11:35:11 OCA PAD INITIATION - PROJECT HEADER INFORMATION 08/23/88

Active

Project #: E-21-F21
Center # : R6571-0A0

Cost share #:
Center shr #:

Rev #: 0
OCA file #:
Work type : RES
Document : CONT
Contract entity: GTRC

Contract#: MDA970-88-C-8145
Prime #:

Mod #:

Subprojects ? : N
Main project #:

Project unit: EE Unit code: 02.010.118
Project director(s):
HERTLING D R EE
FEENEY R K EE
JOY E B EE

Sponsor/division names: US DEPT OF DEFENSE / MARYLAND PROCUREMENT OFF
Sponsor/division codes: 101 / 010

Award period: 880628 to 890627 (performance) 890627 (reports)

Sponsor amount	New this change	Total to date
Contract value	290,666.00	290,666.00
Funded	290,666.00	290,666.00
Cost sharing amount		0.00

Does subcontracting plan apply ?: N

Title: PLANAR SURFACE, DIPOLE, PHASED ARRAY COMPUTER AIDED DESIGN AND LAYOUT PROG.

PROJECT ADMINISTRATION DATA

OCA contact: John B. Schonk 894-4820

Sponsor technical contact

JOE ZIMMERMAN
(301)688-8141
MARYLAND PROCUREMENT OFFICE
9800 SAVAGE RAOD
FT. GEORGE G. MEADE, MD 20755-6000

Sponsor issuing office

GREGORY A. FREAM/ L462
(301)688-5482

SAME

Security class (U,C,S,TS) : U
Defense priority rating : DO-A7
Equipment title vests with: Sponsor X
IN ACCORDANCE WITH FAR 52.245-5, ALT 1

ONR resident rep. is ACO (Y/N): N
GOVT supplemental sheet
GIT

Administrative comments -
PROJECT INITIATION

GEORGIA INSTITUTE OF TECHNOLOGY
OFFICE OF CONTRACT ADMINISTRATION

NOTICE OF PROJECT CLOSEOUT

Closeout Notice Date 04/30/90

Project No. E-21-F21 Center No. R6571-0A0

Project Director HERTLING D R School/Lab EE

Sponsor US DEPT OF DEFENSE/MARYLAND PROCUREMENT OFF

Contract/Grant No. MDA904-88-C-8145 Contract Entity GTRC

Prime Contract No.

Title A PLANAR SURFACE, DIPOLE, PHASED ARRAY COMPUTER AIDED DESIGN AND LAYOUT P

Effective Completion Date 890930 (Performance) 890930 (Reports)

Closeout Actions Required:	Y/N	Date Submitted
Final Invoice or Copy of Final Invoice	Y	
Final Report of Inventions and/or Subcontracts	Y	
Government Property Inventory & Related Certificate	Y	
Classified Material Certificate	N	
Release and Assignment	Y	
Other	N	

Comments

Subproject Under Main Project No.

Continues Project No.

Distribution Required:

Project Director	Y
Administrative Network Representative	Y
GTRI Accounting/Grants and Contracts	Y
Procurement/Supply Services	Y
Research Property Managment	Y
Research Security Services	N
Reports Coordinator (OCA)	Y
GTRC	Y
Project File	Y
Other	N
	N

NOTE: Final Patent Questionnaire sent to PDPI.

E-21-F21

SR-300

A PLANAR SURFACE, DIPOLE, PHASED ARRAY COMPUTER AIDED DESIGN AND LAYOUT PROGRAM

Monthly Status Report

For the periods

7/1/88 - 8/1/88
and
8/2/88 - 9/1/88

Contract Number: MDA970-88-C-8145

Prepared by

Georgia Institute of Technology
School of Electrical Engineering
Atlanta, Georgia

SUMMARY

During the reporting period, the project has been initialized and progress has been made on the Antenna Design and Analysis Task and on the Feed and Filter Design Task. This report covers the first two months of the project. Accomplished work includes literature searches and preliminary calculations of antenna self- and mutual-impedances and prototype feed configurations and circuits. Various copper foil tape on mylar transmission line structures are currently being constructed and tested. Currently, an antenna is being designed which will be constructed and tested. The project is on schedule.

TASK STATUS

During the period covered by this report, efforts have been concentrated on project initialization, literature searches, and preliminary calculations. The project is currently on schedule. We have met with Mr. Joe Zimmerman on two occasions and have reported our efforts and have discussed planned efforts toward completion of the project goals and objectives. Work has been accomplished in two of the three major tasks, the Antenna Design and Analysis Task and the Feed and Filter Design Task. A summary of activity is presented below.

Antenna Design and Analysis Task

1. A literature search has been conducted to find equations and computation techniques for the calculation of self- and mutual-impedance of dipoles. Three results were obtained.

   A. The Numerical Electromagnetic Code (NEC) is a computer algorithm which uses the method of moments to solve for the current distributions and self- and mutual-impedances of wire structures, such as dipoles. The code is quite lengthy and thus requires a very large amount of computer time and memory to solve a large array of dipoles.

   B. Equations were found for the self- (R. W. P. King, Kraus, and others) impedance of circular cylindrical dipoles and the mutual impedance (Howard King) of infinitely thin, parallel, center-fed dipoles. The equations are very lengthy and contain terms which are integrals.

   C. Simplified equations for infinitely thin, widely spaced, dipoles have also been found. The accuracy of these equations is unknown for the required spacing of approximately one-half wavelength used in this application.

2. The NEC code has been obtained and the key self- and mutual-impedance portions of the code have been extracted. The code has been translated from BASIC to Fortran V computer language. The code has been tested for various dipole

spacings and locations and results are in agreement with the mutual impedance graphs given in the Kraus and Howard King papers.

3. An array analysis methodology has been formulated which uses the power of the method of moments NEC code yet reduces the required computer memory and run time requirements. The array will be composed of identical dipoles: identical in length and identical in diameter (width of tape). The array will be laid out on a regularly spaced rectangular lattice. The self-impedance of the dipoles is calculated with the NEC code assuming they all are equal, thus only one self-impedance is calculated. The mutual impedances between dipoles is calculated between each pairs of dipoles using the NEC code. Since the mutual impedance between any two dipoles in the array which have the same geometrical relationship is the same, the number of calculations and computer storage requirements can be reduced from a number proportional to the square of the number of dipoles to a number proportional to the number of dipoles.

4. Work is underway to develop the array design program which specifies the number of dipoles, the rectangular lattice spacing, the dipole currents, the dipole lengths, and the dipole widths (diameters) required to achieve the desired main beam direction(s) with maximum gain. The program interfaces between the user of the program and outputs information to be used by the feed, matching and filter program and to the array layout program. Several test cases have been designed and are currently being tested against known solutions.

Feed and Filter Design Task

1. A literature search has been conducted to find design and analysis equations for various transmission line structures which can be easily constructed with copper foil tape on mylar. Several different sets of equations have been found.

2. Current work includes the construction of transmission lines on mylar and the measurement of their characteristic impedances and velocity factors to check the accuracy of the describing equations.

3. Various feed configurations for the antenna columns are being designed and evaluated. These networks provide the magnitude and phase for the antenna currents for the array.

FUTURE PLANS

During the next reporting period, work will continue on the antenna impedance calculations and on the design of the feed networks. More measurements will be made of the characteristic impedance and velocity factor of various copper foil tape on mylar transmission lines. Work on the design, construction, and testing of a prototype antenna will begin.

EXPENDITURES

Personal Services	
D. R. Hertling, Professor	$ 10,500
E. B. Joy, Professor	$ 8,978
R. E. Wilson, Electronic Eng.	$ 4,867
Michael Guler, Grad. Res. Asst.	$ 2,160
David Camp, Grad. Res. Asst.	$ 1,860
Fringe Benefits (25.5% of personal services excluding grad. strudents)	$ 6,208
Materials and Supplies	
Copper foil tape	$ 132
Computer programs	$ 200
Overhead (60% of direct costs)	$ 20,943
Total Expenditures	$ 55,848
Remaining funds as of 9/1/88	$234,818

E-21-F-21

A PLANAR SURFACE, DIPOLE, PHASED ARRAY COMPUTER
AIDED DESIGN AND LAYOUT PROGRAM

Monthly Status Report

For the period

9/1/88 -10/1/88

Contract Number: MDA970-88-C-8145

Prepared by

Georgia Institute of Technology
School of Electrical Engineering
Atlanta, Georgia

SUMMARY

During the reporting period, progress has continued on the Antenna Design and Analysis Task and on the Feed and Filter Design Task. Accomplished work includes progress on the software which calculates antenna self- and mutual-impedances and designs the antenna array. Construction and testing of various copper tape on mylar transmission line structures continued. The project is on schedule.

We have not yet received the government furnished property as per section H.7A of the contract. In particular, we need the computers and plotter to begin testing the software. We will also need the amplifiers as soon as possible to test in prototype antenna arrays.

TASK STATUS

During the period covered by this report, efforts have been concentrated on the refinement of the antenna design software and the design and testing of copper foil on mylar transmission line structures. Work has been accomplished in two of the three major tasks, the Antenna Design and Analysis Task and the Feed and Filter Design Task. Some preliminary literature search work has been accomplished on the layout and routing task. The project is currently on schedule. A summary of activity is presented below.

Antenna Design and Analysis Task

1. The antenna design and analysis computer program main framework has been completed. The work now centers on development of the following main components.

    1. Array size program.
    2. Array spacing program.
    3. Pointing geometry program.
    4. Element length program.
    5. Element polarization program.
    6. Element width program.
    7. Self-impedance program.
    8. Mutual-impedance program.
    9. Matrix inversion program.
    10. User interface program.
    11. Point source far-field pattern program.
    12. Element pattern program.
    13. Far-field polarization program.
    14. Element excitation program.
    15. Data output program to the feed and filter program and the automated plotting program.

Programs 1-10 are approximately 70% complete and are being tested and packaged. Several versions of the key self- and mutual-impedance programs are being tested against each other.

A simple test case of a broadside array has been designed and will be constructed.

Feed and Filter Design Task

1. Copper foil on mylar transmission line structures which can be implemented on one side of the mylar and structures which can be implemented on both sides of the mylar have been analyzed. Several of these transmission lines have been constructed and measurements of characteristic impedance and velocity factor have been made.

2. Considerable effort has been spent on evaluation of different feed and filter networks. Various combinations of matching networks and splitter/combiner networks are being considered. Protoype feed and filter networks are currently being designed to be used on a prototype antenna aray.

FUTURE PLANS

During the next reporting period, work will continue on the antenna impedance calculations and on the design of the feed networks. Work on the design, construction, and testing of a prototype antenna will continue.

EXPENDITURES

Personal Services

D. R. Hertling, Professor	$	5,250
E. B. Joy, Professor	$	4,489
R. E. Wilson, Electronic Eng.	$	2,496
Michael Guler, Grad. Res. Asst.	$	1,080
David Camp, Grad. Res. Asst.	$	930
Fringe Benefits (25.5% of personal services excluding grad. strudents)	$	3,120

Materials and Supplies

Copper foil tape	$	155
Computer programs	$	6
Overhead (60% of direct costs)	$	10,516
Total Expenditures	$	28,042
Remaining funds as of 9/1/88	$	234,818

E-21-F21

A PLANAR SURFACE, DIPOLE, PHASED ARRAY COMPUTER AIDED DESIGN AND LAYOUT PROGRAM

Monthly Status Report

For the period

10/1/88 -10/31/88

Contract Number: MDA970-88-C-8145

Prepared by

Georgia Institute of Technology
School of Electrical Engineering
Atlanta, Georgia

SUMMARY

During the reporting period, progress has continued on the Antenna Design and Analysis Task and on the Feed and Filter Design Task. Accomplished work includes progress on the software which calculates antenna self- and mutual-impedances and designs the antenna array. In particular, the speed of this program has been greatly increased and the capability to automatically select copper foil widths has been added. Construction and testing of various copper tape on mylar transmission line structures continued. The project is on schedule.

We have not yet received the government furnished property as per section H.7A of the contract. In particular, we need the computers and plotter to begin testing the software. We will also need the amplifiers as soon as possible to test in prototype antenna arrays.

TASK STATUS

During the period covered by this report, efforts have been concentrated on the refinement of the antenna design software and the design and testing of copper foil on mylar transmission line structures. Work has been accomplished in two of the three major tasks, the Antenna Design and Analysis Task and the Feed and Filter Design Task. Some preliminary literature search work has been accomplished on the layout and routing task. The project is currently on schedule. A summary of activity is presented below.

Antenna Design and Analysis Task

1. The computation time was greatly reduced by taking advantage of the symmetry in the mutual-impedance matrix. There are N x N impedance (self and mutual) entries in the impedance matrix for an N element array. Taking full advantage of the matrix symmetry reduces this number to N for a rectangularly spaced array of equal sized dipoles, such as ours. For a 25 element array, the number of entries (and associated computation time) decreases from 625 to 25, a factor of 25. Computation time for a 12 element array is now 12 seconds on the campus mainframe and 12 minutes on an IBM-XT.

2. The width of the copper foil tape used to make the dipole antennas is now automatically selected from a list of standard widths.

Feed and Filter Design Task

1. Copper foil on mylar transmission line structures which can be implemented on one side of the mylar and structures which can be implemented on both sides of the mylar have been analyzed. Several of these transmission lines have been constructed and measurements of characteristic impedance and velocity factor have been made.

2. Considerable effort has been spent on evaluation of different feed and filter networks. Various combinations of matching networks and splitter/combiner networks are being considered. Protoype feed and filter networks are currently being designed to be used on a prototype antenna array.

FUTURE PLANS

During the next reporting period, work will continue on the antenna impedance calculations and on the design of the feed networks. In particular, the capability to automatically calculate the horizontal and vertical spacings of the array dipoles will be added. Work on the design, construction, and testing of a prototype antenna will continue.

EXPENDITURES

Personal Services	
E. B. Joy, Professor	$ 1,901
R. E. Wilson, Electronic Eng.	$ 2,496
Michael Guler, Grad. Res. Asst.	$ 1,080
David Camp, Grad. Res. Asst.	$ 930
Fringe Benefits (25.5% of personal services excluding grad. students)	$ 1,121
Materials and Supplies	$ 360
Overhead (60% of direct costs)	$ 4,732
Total Expenditures	$ 12,621
Remaining funds as of 10/30/88	$194,155

E21-F21

A PLANAR SURFACE, DIPOLE, PHASED ARRAY COMPUTER AIDED DESIGN AND LAYOUT PROGRAM

Monthly Status Report

For the period

11/1/88 -11/30/88

Contract Number: MDA970-88-C-8145

Prepared by

Georgia Institute of Technology
School of Electrical Engineering
Atlanta, Georgia

SUMMARY

During the reporting period, progress has continued on the Antenna Design and Analysis Task and on the Feed and Filter Design Task. The capability to automatically calculate horizontal and vertical spacings of the array dipoles has been added to the program which calculates the mutual impedances of the array dipoles. A new self- and mutual-impedance algorithm has also been developed. The project is on schedule.

We have not yet received the government furnished property as per section H.7A of the contract. In particular, we need the computers and plotter to begin testing the software. We will also need the amplifiers as soon as possible to test in prototype antenna arrays.

TASK STATUS

During the period covered by this report, efforts have been concentrated on the refinement of the antenna design software and the design and testing of copper foil on mylar transmission line structures. Work has been accomplished in all three major tasks. The literature search for routing and placement algorithms has continued. Additional references have been found and evaluated. The project is currently on schedule. A summary of activity is presented below.

Antenna Design and Analysis Task

1. The horizontal and vertical spacings of the array dipoles are now automatically determined such that the widest possible spacings are used for maximum gain, but less than that which would produce a grating lobe.

2. A new self- and mutual impedance algorithm has been developed, implemented and partially tested based on the equations of H. E. King. The calculated self- and mutual-impedances of and between parallel and offset dipoles is being tested against data measured by R. W. P. King. This new form has great computational advantages over the MININEC code previously used. Work continues in the testing and in the calculation of the mutual impedance of electrically thick dipoles, such as the dipoles we are using.

Feed and Filter Design Task

Work continued on the evaluation of feed and filter networks. Additional feed and filter network designs have been considered.

Routing and Layout Design Task

The literature search for existing layout algorithms continued. One program which uses a simple routing algorithm has been acquired and preliminary testing on an IBM-XT as performed.

FUTURE PLANS

During the next reporting period, work will continue on the antenna design software. The ability to tilt the array face in elevation will be added. Calculation and display of azimuth cuts and elevation cuts for the far-field pattern of the array antenna will be added to the program. The accuaracy of the new self- and mutual-impedance algorithm will be evaluated. Work on the design, construction, and testing of a prototype antenna will also continue.

EXPENDITURES

Personal Services	
E. B. Joy, Professor	$ 1,901
R. E. Wilson, Electronic Eng.	$ 2,496
Michael Guler, Grad. Res. Asst.	$ 1,080
David Camp, Grad. Res. Asst.	$ 930
Fringe Benefits (25.5% of personal services excluding grad. students)	$ 1,121
Materials and Supplies	$ 106
Overhead (60% of direct costs)	$ 4,580
Total Expenditures	$ 12,214
Remaining funds as of 12/1/88	$181,941

E-21-F2

A PLANAR SURFACE, DIPOLE, PHASED ARRAY COMPUTER AIDED DESIGN AND LAYOUT PROGRAM

Monthly Status Report

For the period

12/1/88 - 12/31/88

Contract Number: MDA970-88-C-8145

Prepared by

Georgia Institute of Technology
School of Electrical Engineering
Atlanta, Georgia

SUMMARY

During the reporting period, progress has continued on the Antenna Design and Analysis Task and on the Feed and Filter Design Task. In particular, the ability to tilt the array face in elevation has been implemented and azimuth and elevation cuts of the far-field pattern of the array antenna are now calculated and displayed. The new self- and mutual-impedance algorithm has been tested. The project is on schedule.

We have received part of the government furnished property as per section H.7A of the contract. We have received the computer and plotter. We have not yet received the magnetic media for the hard disk or the amplifiers.

On December 14, 1988 we met at Georgia Tech with Joe Zimmerman. During the meeting a review of the project was given and goals for the next month were discussed.

TASK STATUS

During the period covered by this report, efforts have been concentrated on the refinement of the antenna design software and the design and testing of copper foil on mylar transmission line structures. The literature search for routing and placement algorithms has continued. Additional references have been found and evaluated. Work has been accomplished in all of the three major tasks and the project is currently on schedule. A summary of activity is presented below.

Antenna Design and Analysis Task

1. The ability to tilt the array face in elevation has been implemented. The user specifies the angular tilt in degrees.

2. Azimuth cuts (for a fixed elevation angle) of the far-field pattern of the array antenna and elevation cuts (for a fixed azimuth angle) of the far-field pattern of the array antenna are now calculated and displayed for horizontally oriented infinitesimal dipole element arrays.

3. The new self- and mutual-impedance algorithm has been tested and found to give accurate results for dipoles with thicknesses over a range of two orders of magnitude, but not as large as needed in the required array. Initial work on representing a single fat wire with 2, 4, or 8 slender wires looks very promising and is being pursued. Accuracy studies are also underway to examine the effect of length to diameter ratio and number of segments on the accuracy of self-impedance determination.

Feed and Filter Design Task

Work continued on the evaluation of feed and filter networks. Additional feed and filter network designs have been considered.

Routing and Layout Design Task

The literature search for existing layout algorithms continued. Preliminary considerations of how to interface with the plotter have been considered.

FUTURE PLANS

During the next reporting period, work will continue on the antenna design software. Work will continue to upgrade the capability of computing and displaying the azimuth and elevation cuts for full length dipoles with arbitrary orientation on the tilted array face.

EXPENDITURES

Personal Services	
E. B. Joy, Professor	$ 1,901
R. E. Wilson, Electronic Eng.	$ 2,246
Michael Guler, Grad. Res. Asst.	$ 1,080
David Camp, Grad. Res. Asst.	$ 930
Fringe Benefits (25.5% of personal services excluding grad. students)	$ 1,057
Materials and Supplies	$ 227
Computer Software	$ 680
Overhead (60% of direct costs)	$ 4,873
Total Expenditures	$ 12,994
Remaining funds as of 1/1/89	$168,947

E-21-F21

A PLANAR SURFACE, DIPOLE, PHASED ARRAY COMPUTER AIDED DESIGN AND LAYOUT PROGRAM

Monthly Status Report

For the period

1/1/89 -1/31/89

Contract Number: MDA970-88-C-8145

Prepared by

Georgia Institute of Technology
School of Electrical Engineering
Atlanta, Georgia

SUMMARY

During the reporting period, progress has continued on all tasks of the project. The self impedance algorithm has been successfully implemented on the personal computer. Accuracy studies have shown the self impedance of dipoles with lengths from 0.3 to 0.5 wavelengths can be accurately computed using our method of moments algorithm. A feed and filter network using only transmission lines and capacitors has been designed. The project is on schedule.

TASK STATUS

During the period covered by this report, efforts have been concentrated on the refinement of the antenna design software and the design of feed and filter networks. Work has been accomplished in all of the three major tasks and the project is currently on schedule. A summary of activity is presented below.

Antenna Design and Analysis Task

1. Implementation of the self impedance algorithm on an IBM-XT computer resulted in a numerical problem associated with the small mumber of significant digits used in such microcomputers. The numerical problem was solved using a closed form equation for the unstable computation. The self impedance algorithm is now running successfully on the microcomputer.

2. Accuracy studies on the self impedance show that the self impedance of dipoles with length in the range of 0.3 to 0.5 wavelengths can be accurately computed using our method of moments algorithm. The accurracy has been compared to R. W. P. King's published data and to the NEC algorithm. Our method of moments algorithm has a great computational speed advantage as compared to the NEC algorithm.

3. Work continues on the calculation of the self impedance of the electrically large diameter dipoles.

4. Work continues on the accuracy of mutual impedance calculation using our method of moments algorithm.

Feed and Filter Design Task

1. A feed and filter network containing only transmission lines and capacitors has been designed. This network will be much easier to construct than a network containing both inductors and capacitors. Another advantage is lower loss since only high Q capacitors are used. Routing and placement will also be simplified with this feed network.

2. A survey of the various thicknesses of mylar sheets which are readily available has been made. We have also learned that Mylar is available in 10 foot rolls which makes it possible to fabricate many of the antennas without seams.

FUTURE PLANS

During the next reporting period, work will continue on the calculation of the self impedance of electrically large diameter dipoles and on improvement of the accuracy of the method of moments algorithm. Further refinement of the design procedures for the feed and filter network will also continue. The first design was performed using manual optimization of the feed network parameters. An optimizer is currently being written to automatically optimize the feed and filter network. An antenna using the first feed and filter network design will be constructed.

EXPENDITURES

Personal Services

R. K. Feeney, Professor	$ 4,493
E. B. Joy, Professor	$ 2,333
R. E. Wilson, Electronic Eng.	$ 2,246
Michael Guler, Grad. Res. Asst.	$ 1,080
David Camp, Grad. Res. Asst.	$ 930
Fringe Benefits (25.5% of personal services excluding grad. students)	$ 2,313
Materials and Supplies	$ 0
Computer Software	$ 0
Overhead (60% of direct costs)	$ 8,037
Total Expenditures	$ 21,432
Remaining funds as of 2/1/89	$147,515

E-21-F21

A PLANAR SURFACE, DIPOLE, PHASED ARRAY COMPUTER AIDED DESIGN AND LAYOUT PROGRAM

Monthly Status Report

For the period

3/1/89 -3/31/89

Contract Number: MDA970-88-C-8145

Prepared by

Georgia Institute of Technology
School of Electrical Engineering
Atlanta, Georgia

SUMMARY

During the reporting period, progress has continued on all tasks of the project. The method of moments algorithm is now running on the government furnished PC and several improvements to the algorithm have been mad to improve accuracy. A new technique for the numerical evaluation of mutual impedance has been developed which does not require the use of a load impedance on one of the two dipoles. A program which deterministically designs matching networks rather than using an optimizer has been written. Work on the layout and plotting task has also been accomplished. The project is on schedule.

TASK STATUS

During the period covered by this report, efforts have concentrated on the refinement of the antenna design software and the design of feed and filter networks. Work has been accomplished in all of the three major tasks and the project is currently on schedule. A summary of activity is presented below.

Antenna Design and Analysis Task

1. Several of the sine and cosine integral routines and associated approximate series have been modified to obtain the desired accuracy using 16 bit words (word length of the PC).

2. Accuracy studies have been successfully carried out to verify the PC version accuracy.

3. Accuracy studies of the self impedance determinations as compared to measured data of R. W. P. King show that high accuracy is obtained for the following lengths and radii of dipole antennas:

Length:	0.3 to 0.5 wavelengths
Radius:	0.000001 to 0.01 wavelengths

   These accuracies are obtained using six segments per dipole. These ranges are of importance and are used in this project.

4. A new technique for the numerical evaluation of mutual impedance has been developed which does not require the use of a load impedance on one of the two dipoles. This load impedance must be set to a very large value and is a source of some numerical instability. Without the need for this load impedance numerical results are shown to be highly stable. The effect of dipole thickness can now be more easily calculated using the new formulation.

Feed and Filter Design Task

1. It was found that the feed and filter network containing only transmission lines and capacitors would not work for all antenna impedances and electrical line lengths. A computer program which deterministically designs matching and phasing networks for the dipole antennas has been written.

2. The construction of lumped capacitors and inductors using the copper foil tape are being developed. Both series and shunt capacitors and inductors will be needed in general. Methods of interfacing the feed and filter design software with the layout and plotting software are being developed.

3. We are also in the process of designing standard T and PI networks using copper foil tape on MYLAR. Once these have been designed, any T and PI network needed can be easily realized.

FUTURE PLANS

During the next reporting period work will continue on all tasks. In particular, work will concentrate on the building and testing antennas. The interfacing of the various computer programs will also begin. That is, the self and mutual impedance algorithm must pass its output to the feed and filter design program which must then pass data on to the routing and layout program.

EXPENDITURES

Personal Services	
R. K. Feeney, Professor	$ 4,493
E. B. Joy, Professor	$ 2,333
R. E. Wilson, Electronic Eng.	$ 2,246
Michael Guler, Grad. Res. Asst.	$ 1,080
David Camp, Grad. Res. Asst.	$ 930
Fringe Benefits (25.5% of personal services excluding grad. students)	$ 2,313
Materials and Supplies	$ 0
Computer Software	$ 0
Overhead (60% of direct costs)	$ 8,037
Total Expenditures	$ 21,432
Remaining funds as of 4/1/89	$104,651

E-21-F21

A PLANAR SURFACE, DIPOLE, PHASED ARRAY COMPUTER AIDED DESIGN AND LAYOUT PROGRAM

Monthly Status Report

For the period

4/1/89 - 4/30/89

Contract Number: MDA970-88-C-8145

Prepared by

Georgia Institute of Technology
School of Electrical Engineering
Atlanta, Georgia

SUMMARY

During the reporting period, progress has continued on the Antenna Design and Analysis Task and on the Feed and Filter Design Task. Work was also accomplished on the Routing and Layout Design Task. In particular, improvements have been made to the software which calculates antenna impedances and to the software which designs the feed and filter networks. The project is on schedule.

TASK STATUS

During the period covered by this report, efforts have been concentrated on the refinement of the antenna design software, the feed and filter design software and on the interfacing of the these programs. The feed and filter design program has been improved and its results checked with a circuit simulator to verify that the feed and filter networks are correct. The writing of the routing and placement software has begun. Work has been accomplished in all of the three major tasks and the project is currently on schedule. A summary of activity is presented below.

Antenna Design and Analysis Task

1. An error in the computation of the dipole length was corrected. This occurred if the dipole tilt put the end of the dipole in any quadrant except quadrant one.

2. The wording of the prompt for dipole tilt was changed to include a description of what the program considers positive. A positive dipole tilt angle is considered to be in a counter-clockwise direction when looking in the same direction as the array.

3. The prompt for location of the filter network was changed so that it is dependant on the dipole tilt angle. For

$$0^{o} < \text{Dipole Tilt} < 45^{o}$$

only left or right are allowed positions. For

$$45^{o} \leq \text{Dopole tilt} < 90^{o}$$

only top or bottom are allowed positions. Here the angle supplied by the user has been adjusted so that dipole tilt is between 0^{o} and 90^{o}.

4. It is possible that the desired combination of frequency and panel size will result in more elements being required than can be handled by the program. The error handling for this condition was changed so that the user is apprised of the situation and asked to specify a smaller panel size. Previously, the program generated an error message and aborted.

5. The computation of the driving point impedances was modified so that it is necessary to store only one row/column of the system impedance matrix. This makes it possible to allow the design of up to 20 x 20 element array including a ground plane on the PC.

Feed and Filter Design Task

1. Work continued on the program which deterministically designs the antenna feed networks. This program accepts data from the self and mutual impedances and then designs the antenna feed networks. A circuit simulator was used to verify that the feed networks designed by this program are correct. This program can calculate both balanced T networks and balanced PI networks. Whenever possible, balanced PI network are used since they contain fewer elements and are thus easier to construct.

2. A 491 MHz single column with three dipoles has been constructed. We are currently in the process of making impedance and pattern measurements of this antenna.

3. More work has been accomplished on the practical realization of the various T and PI feed networks. There are a total of 16 (eight T and eight PI) possible feed networks. Work is currently underway to realize each one of these networks using copper foil tape on mylar.

Routing and Layout Design Task

1. The software which accepts data from both the antenna impedance program and the feed network program has been written and is in the process of being de-bugged. This program will route and plot the feed networks and the antennas.

2. Preliminary work has been accomplished to interface this program to the antenna impedance and the feed and filter design programs.

FUTURE PLANS

During the next reporting period, work will concentrate on the completion and the interfacing of the the programs. Work will also continue on the construction and testing of antenna arrays.

EXPENDITURES

Personal Services	
R. K. Feeney, Professor	$ 4,493
D. R. Hertling	$ 3,450
E. B. Joy, Professor	$ 2,519
R. E. Wilson, Electronic Eng.	$ 2,246
Michael Guler, Grad. Res. Asst.	$ 1,080
David Camp, Grad. Res. Asst.	$ 930
Joseph Epple, Grad. Res. Asst.	$ 720
Fringe Benefits (25.5% of personal services excluding grad. students)	$ 3,241
Materials and Supplies	$ 0
Computer Software	$ 0
Overhead (60% of direct costs)	$ 11,207
Total Expenditures	$ 29,886
Remaining funds as of 5/1/89	$ 74,765

E-21-F21

A PLANAR SURFACE, DIPOLE, PHASED ARRAY COMPUTER AIDED DESIGN AND LAYOUT PROGRAM

Monthly Status Report

For the period

5/1/89 - 5/31/89

Contract Number: MDA970-88-C-8145

Prepared by

Georgia Institute of Technology
School of Electrical Engineering
Atlanta, Georgia

SUMMARY

During the reporting period, progress has continued on the Antenna Design and Analysis Task, on the Feed and Filter Design Task, and on the Routing and Layout Design Task. In particular, improvements have been made to all three programs and to interfacing them. On May 11 and May 12 we met at Georgia Tech with Mr. Joe Zimmerman and Dr. Bob Collier. During their visit, several antennas were designed using the software developed on the project. Bob Collier is currently building and testing several antennas which have been designed with the software. We are also working on developing the capability to etch antennas using conventional printed circuit board techniques.

TASK STATUS

During the period covered by this report, efforts have been concentrated on the refinement of the antenna design software, the feed and filter design software and on the interfacing of the these programs. Work has been accomplished in all of the three major tasks and the project is currently on schedule. A summary of activity is presented below.

Antenna Design and Analysis Task

1. Handling of a ground plane was added to the program. The user is asked if a ground plane is desired. If a ground plane is desired, the program computes the required spacing between the array panel and the ground plane.

2. If a ground plane is requested, the user is also asked for the value of the dielectric constant for the spacer material. This is used in the computation of the spacing between the array panel and the ground plane.

3. The list of standard tape widths used in determining the dipole width was changed to readily available, off the shelf values. The current tape widths used by the program are 0.125", 0.25", 0.375", 0.5", 0.75", and 1.0" .

4. Interface routines were added to prompt for the dielectric constant and thickness of the Mylar, and to write the design parameters to disk. This disk file is used by the Feed and Filter Network and Routing and Placement programs.

5. "Please Wait" messages were added during the computation of the driving point impedances and the generation of pattern plots to assure the user that the program is executing and has not "locked up". Also, if plots are to be sent to the HP plotter, the program prompts the user to load paper and waits for user confirmation.

Feed and Filter Design Task

1. The Feed and Filter design program now automatically reads data from the file generated by the Antenna Design and Analysis program and designs the feed networks for all dipoles.

2. Impedance measurements on the 491 MHz single column with three dipoles have been made. These impedances differ from the predicted values and we are currently in the process of resolving the discrepency.

3. More work has been accomplished on the practical realization of the various T and PI feed networks. All of the basic networks have been laid out. Work is currently underway to measure the various series and shunt inductors and capacitors required for the networks.

4. Work has been accomplished on the interfacing of the Feed and Filter design program to the Placement and Routing program.

Routing and Layout Design Task

1. The capability has been added to meander the transmission lines in order to change their physical lengths. This provides the capability to set the electrical length between dipoles.

2. We have begun to incorporate the various T and PI networks into the Routing and Layout program.

FUTURE PLANS

During the next reporting period, work will concentrate on the completion and the interfacing of the the programs. Work will also continue on the construction and testing of antenna arrays. We will also continue to design antennas specified by Joe Zimmerman.

EXPENDITURES

Personal Services	
R. K. Feeney, Professor	$ 4,493
D. R. Hertling	$ 3,450
E. B. Joy, Professor	$ 2,519
R. E. Wilson, Electronic Eng.	$ 2,246
Michael Guler, Grad. Res. Asst.	$ 1,080
David Camp, Grad. Res. Asst.	$ 930
Joseph Epple, Grad. Res. Asst.	$ 720
Fringe Benefits (25.5% of personal services excluding grad. students)	$ 3,241
Materials and Supplies	$ 0
Computer Software	$ 0
Overhead (60% of direct costs)	$ 11,207
Total Expenditures	$ 29,886
Remaining funds as of 6/1/89	$ 44,879

E-21-F21

A PLANAR SURFACE, DIPOLE, PHASED ARRAY COMPUTER AIDED DESIGN AND LAYOUT PROGRAM

Monthly Status Report

For the period

6/1/89 - 6/30/89

Contract Number: MDA970-88-C-8145

Prepared by

Georgia Institute of Technology
School of Electrical Engineering
Atlanta, Georgia

SUMMARY

During the reporting period, progress has continued on the Antenna Design and Analysis Task, on the Feed and Filter Design Task, and on the Routing and Layout Design Task. In particular, improvements have been made to all three programs and to interfacing them. Several antennas have been built and tested. Additionally, we have been working with Dr. Bob Collier who has been building and testing antennas. Additional work on developing the capability to etch antennas using conventional printed circuit board techniques has been accomplished. Mr. Joe Zimmerman, Dr. Bob Collier, and Mr. Richard Wilson visited Buckbee Mears in St. Paul, MN on 8 June to discuss the fabrication of etched antennas on microwave substrate PC board material.

Problems with the lossy adhesive on the copper foil tape were identified and copper foil tape with a low loss adhesive has been obtained. It has also been found that the loss of Mylar is significant in the UHF range. Identification of these problems caused delays in antenna fabrication and testing and a no-cost extension (1 Sep 89) of the contract was requested.

TASK STATUS

During the period covered by this report, efforts have concentrated on the refinement of the antenna design software, the feed and filter design software, the routing and placement software and on the interfacing of these programs. In particular, considerable progress has been made on the routing and antenna layout software. Work has been accomplished in all of the three major tasks and the project is currently on schedule. A summary of activity is presented below.

Antenna Design and Analysis Task

1. An error was found in our method of computing the mutual impedances in the system impedance matrix. The error occurred only when the dipoles were tilted (i.e. when the dipoles were not horizontal or vertical). This error was corrected without a major impact on the computation time or on the maximum size of the array which can be accomodated. The program can still handle a 20 x 20 element array.

2. Since, in the future, the array panel may be something other than Mylar, references to Mylar in the program prompts were changed to "substrate".

3. In addition to the required distance to the ground plane computed by the program, the output printout now includes the value of the spacer dielectric constant which is used in that computation and specified by the user.

4. Drafts of the user's manual for DIPARA were written.

5. The program now writes the data required by the feed and filter design and the routing and layout programs to the current default drive rather than to the A: drive. This facilitates running all three programs from a batch file.

6. A batch file was written to run the array design program, the feed and filter design program, and the routing and layout program with one command from the command line. The batch file method has been tested and works.

Feed and Filter Design Task

1. The feed and filter design program now automatically reads data from the file generated by the antenna design and analysis program and writes to a file to be read by the routing and layout program.

2. The discrepencies in the impedance measurements on the 491 MHz single column with three dipoles have been traced to the loss in the copper foil tape adhesive and in the Mylar. The transmission lines which used the adhesive and the Mylar as a dielectric were replaced with coplanar strip transmission lines and the impedance measurements remade. These impedances agreed much better with the predicted values. The column impedance was approximately 75 + j30 ohms at 491 MHz and was 50 +j0 ohms at 508 MHz.

3. Pattern measurements on the 491 MHz column antenna were made. The measured pattern agreed fairly well with the predicted values. The nulls in the pattern and and sidelobes were approximately at the correct angles but were not the correct levels. We are currently making additional measurements with this antenna.

4. We are currently measuring various series and shunt inductors and capacitors required for the feed networks networks. These measurements are being made on several substrate materials including Mylar, teflon-glass, and epoxy glass. This data on the capacitors and inductors is needed by the routing and placement program.

5. Work has been accomplished on the interfacing of the feed and filter design program to the routing and placement program.

Routing and Layout Design Task

1. The routing and layout program can now plot the dipole array and the antenna feed networks. Additional work needs to be accomplished so that the capacitors and inductors for all the possible feed networks can be automatically plotted.

2. The routing and layout program has been interfaced to the feed and filter design program.

FUTURE PLANS

During the next reporting period, work will concentrate on the completion of the software and its documentation. Work will also continue on the construction and testing of antenna arrays. In particular, both copper foil on Mylar and etched antennas on teflon-glass PC board will be built and tested. We will also continue to design antennas specified by Mr. Joe Zimmerman and will continue to send antenna designs to Dr. Bob Collier.

EXPENDITURES

Personal Services	
R. K. Feeney, Professor	$ 4,493
D. R. Hertling	$ 3,450
E. B. Joy, Professor	$ 2,519
R. E. Wilson, Electronic Eng.	$ 2,246
Michael Guler, Grad. Res. Asst.	$ 1,080
David Camp, Grad. Res. Asst.	$ 930
Joseph Epple, Grad. Res. Asst.	$ 720
Fringe Benefits (25.5% of personal services excluding grad. students)	$ 3,241
Materials and Supplies	$ 26
Travel	$ 850
Overhead (60% of direct costs)	$ 11,733
Total Expenditures	$ 31,288
Remaining funds as of 7/1/89	$ 13,591

E-21-F21

A PLANAR SURFACE, DIPOLE, PHASED ARRAY COMPUTER AIDED DESIGN AND LAYOUT PROGRAM

Monthly Status Report

For the period

7/1/89 - 7/31/89

Contract Number: MDA970-88-C-8145

Prepared by

Georgia Institute of Technology
School of Electrical Engineering
Atlanta, Georgia

SUMMARY

During the reporting period, progress has continued on the Antenna Design and Analysis Task, on the Feed and Filter Design Task, and on the Routing and Layout Design Task. In particular, significant improvements have been made to the layout and routing program and to the interfacing of the three major programs. Several antennas have been built and tested. We have continued to work with Dr. Bob Collier who has been building and testing antennas.

On July 20 and 21, Mr. Joe Zimmerman, Ms. Leslie Mathias, and Mr. Bob Geissler visited Georgia Tech for an project review. We discussed the remainder of the project delivered the two element 1990 MHz antenna to them. Subsequent testing of the antenna showed it to agree very well with theoretical predictions.

TASK STATUS

During the period covered by this report, efforts have concentrated on the refinement of the antenna design software, the feed and filter design software, the routing and placement software and on the interfacing of these programs. In particular, considerable progress has been made on the routing and antenna layout software. Work has been accomplished in all of the three major tasks and the project is currently on schedule. A summary of activity is presented below.

Antenna Design and Analysis Task

1. The programs were demonstrated to Joe Zimmerman, Leslie Mathias, and Bob Geissler who had several suggestions for changes to the program DIPARA. The following are their recommended changes.

    a) Re-ordering of the units in the units prompt at the beginning of the program.
    b) Changing the response to the units prompt from letters to numbers.
    c) Changing the input routines so that a typing mistake would <u>not</u> abort the program.
    d) Staggering the plotting of the array so that they would <u>not</u> overwrite each other.
    e) Changing the prompt for the beam position to more clearly indicate to the user that the azimuth angle and the elevation angle are required and in what order.
    f) Changing the informative message displayed while the pattern cuts are being computed so that the user is not mislead into thinking that the plotter should be doing something when, in fact, it should not.

2. Low level formatting, partitioning, and formatting of a blank Q-pak was demonstrated. The array design programs and required utilities were loaded onto the freshly formatted Q-pak and several designs were performed using that Q-pak as the boot disk.

3. Something happened to the Q-pak referred to in 2) after being transported to Bendix and it would not initialize. Bob Collier sent another blank Q-pak which was formatted and loaded with the required programs and utilities. This disk was duplicated using a Q-pak which was already at Georgia Tech and both were returned to Bendix.

4. A batch file was written which automatically handles low level formatting and partitioning of a Q-pak. This will facilitate the process of setting up new disk packs in the future.

Feed and Filter Design Task

1. The feed and filter design program has been used to design several antennas. Work has continued on its interfacing to the other programs.

2. The 491 MHz single column with three dipoles was sent to Bendix where its impedance was measured. This was the antenna for which we had to remake the transmission lines because of the lossy coplanar strip lines. The measured input impedance was 112 +j6.8 ohms with an SWR of 2.24. They also measured the antenna at 451 and 530 MHz where the SWR dipped and got 1.074 and 1.615, respectively. We had measured the antenna impedance to be approximately 75 + j30 ohms at 491 MHz and 50 +j0 ohms at 508 MHz. Our measurement was not made on a proper range and the antenna impedance was influenced by surrounding objects. The Bendix measurement was made under better conditions and is, therefore, more accurate.

3. Gain measurements on the 491 MHz column antenna were also made at Bendix. The measured pattern agreed fairly well with the predicted values.

4. Considerable progress has been made on characterizing inductors and capacitors required for the feed networks networks. Measurements are being made on several substrate materials including Mylar, teflon-glass, and epoxy glass. From this data, the routing and placement program calculates the physical dimensions of the required capacitors and inductors.

Routing and Layout Design Task

1. The routing and layout program can now plot the dipole array and the antenna feed networks. It has also been modified to plot on different layers to facilitate antenna construction on both Mylar substrates or for etched circuit board construction.

2. The routing and layout program has been made more user friendly.

FUTURE PLANS

During the next reporting period, work will concentrate on the completion of the software and its documentation. Work will also continue on the construction and testing of antenna arrays. In particular, both copper foil on Mylar and etched antennas on teflon-glass PC board will be built and tested. The placement and routing program will be modified to plot antennas too large for one plot on multiple plots to be pieced together. We will also continue to design antennas specified by Mr. Joe Zimmerman and will continue to send antenna designs to Dr. Bob Collier.

EXPENDITURES

Personal Services	
R. E. Wilson, Electronic Eng.	$ 2,381
Fringe Benefits (26.3% of personal services excluding grad. students)	$ 626
Materials and Supplies	$ 846
Overhead (62.5% of direct costs)	$ 2,408
Total Expenditures	$ 6,261
Remaining funds as of 8/1/89	$ 7,330

E-21-F21

A PLANAR SURFACE, DIPOLE, PHASED ARRAY COMPUTER AIDED DESIGN AND LAYOUT PROGRAM

Monthly Status Report

For the period

8/1/89 - 8/31/89

Contract Number: MDA970-88-C-8145

Prepared by

Georgia Institute of Technology
School of Electrical Engineering
Atlanta, Georgia

SUMMARY

During the reporting period, progress has continued on the Antenna Design and Analysis Task, on the Feed and Filter Design Task, and on the Routing and Layout Design Task. In particular, significant improvements have been made to the layout and routing program and to the interfacing of the three major programs. Several antennas have been built and tested. We have continued to work with Dr. Bob Collier who has been building and testing antennas. A delay in the work was caused by an equipment failure on our HP-8510 network analyzer.

TASK STATUS

During the period covered by this report, efforts have concentrated on the refinement of the antenna design software, the feed and filter design software, the routing and placement software and on the interfacing of these programs. In particular, considerable progress has been made on the routing and antenna layout software. Work has been accomplished in all of the three major tasks and the project is currently on schedule for completion by 1 October. A summary of activity is presented below.

Antenna Design and Analysis Task

1. A routine was added which improves the prompting for the name of the file to contain the design summary. The routine first checks the command line for the name. If the name is not found there, it prompts the user for the name. This capability was previously provided by the FORTRAN library routines, but the prompt was <u>not</u> very informative - except to the programmer. In addition, the new routine checks for the existence of the specified file. The program informs the user if a file already exists and asks for permission to overwrite the file. This provides some protection from inadvertently destroying an important file.

2. The writing of the design summary was moved from the main program to a subroutine to improve the modularity of the program.

3. Code was added to allow the user to check his inputs before the program continues with the design. If the displayed values for the inputs are not correct, the user can re-enter the inputs until satisfied with them.

Feed and Filter Design Task

1. The feed and filter design program was modified to automatically change the phase shift of the matching networks to yield more easily realizable networks.

2. Work on the characterization of the feed networks was interrupted due to an equipment failure of the HP 8510 network analyzer. The analyzer had to be shipped to Texas for repair.

Routing and Layout Design Task

1. The routing and layout software was modified to draw meander lines with smooth corners. This facilitates the construction of etched antennas.

2. The software which plots the layouts for the matching networks was added to the program. Finishing the coding of the networks depends on network analyzer measurements.

3. Several minor software bugs were found and corrected.

FUTURE PLANS

During the next reporting period, work will concentrate on the completion of the software and its documentation. The remaining deliverable antennas will be built and sent to Bendix for testing. The placement and routing program will be modified to plot antennas too large for one plot on multiple plots to be pieced together.

EXPENDITURES

Personal Services		
R. E. Wilson, Electronic Eng.	$	2,381
Fringe Benefits (26.3% of personal services excluding grad. students)	$	626
Materials and Supplies	$	0
Overhead (62.5% of direct costs)	$	1,879
Total Expenditures	$	4,886
Remaining funds as of 9/1/89	$	2,444

E-21-F21

A PLANAR SURFACE, DIPOLE, PHASED ARRAY COMPUTER AIDED DESIGN AND LAYOUT PROGRAM

Monthly Status Report

For the period

9/1/89 - 9/30/89

Contract Number: MDA970-88-C-8145

Prepared by

Georgia Institute of Technology
School of Electrical Engineering
Atlanta, Georgia

SUMMARY

During the reporting period, work has concentrated on finishing the three tasks and completing the project. Several small software changes and corrections have been made. The measurements needed to finish the feed networks have been completed. The project has been completed.

TASK STATUS

During the period covered by this report, efforts have concentrated on de-bugging and finishing the antenna design software, the feed and filter design software. The three tasks have been completed.

Antenna Design and Analysis Task

1. During Mr. Joe Zimmerman's visit on September 14 - 15, several changes were made to the program prompts at his request.
   a) The prompt for the dipole tilt was improved by mentioning **polarity**, indicating the reason for tilting the dipoles was to adjust polarization.

   b) Since polystyrene is a commonly used spacer material between the array panel and a ground plane, the dielectric constant for polystyrene is now included in the prompt for the dielectric constant of the spacer material.

   c) Two common materials used for the array substrate are Teflon glass and Mylar. For the user's convenience, the dielectric constants of these two materials are included in the prompt for the substrate dielectric constant.

2. The user's guide has been updated to include these changes and those described in the July 1989 monthly report. Joe Zimmerman was given a copy of the new user's guide so that he could photocopy it and distribute it during his demonstration of the program.

3. The new versions of the programs were put on two new Q-paks supplied by Joe Zimmerman. DOS and the necessary utilities were also loaded and the Q-paks were made to be able to boot the computer.

Feed and Filter Design Task

1. The final necessary measurements for the completion of the feed networks were made using the network analyzer.

Routing and Layout Design Task

1. The routing and layout software was modified to draw large antennas in several smaller pieces which can be assembled. A smaller scaled drawing is also provided to assist in the assemble of the pieces into the total antenna.

2. The routing and layout software was modified to be made more user-friendly. In particular, either manual or auto-scaling of the plots can now be selected by the user.

3. Software bugs found by Joe Zimmerman in the plotting software were found and corrected.

EXPENDITURES

Personal Services	
R. E. Wilson, Electronic Eng.	$ 1,191
Fringe Benefits (26.3% of personal services excluding grad. students)	$ 313
Materials and Supplies	$ 0
Overhead (62.5% of direct costs)	$ 940
Total Expenditures	$ 2,444
Remaining funds as of 10/1/89	$ 0

E-21-F21

DIPARA: a strip dipole array design program

User's Guide

Table of Contents

Introduction

DIPARA is a strip dipole array design program. The user inputs the source frequency, main beam directions, array panel tilt, dipole tilt, location of a filter network, ground plane information, size of the array panel, and panel substrate dielectric constant and thickness. From this information the program determines the number of dipoles, their locations, and the driving point currents required to produce the desired beam direction. Also, the driving point impedances are determined for use in design of the matching and filter networks.

This user's manual describes how to run the program. To this end the reader is taken step by step through an example design.

Running the Program: an example design

The example design is for a single beam antenna with the beam broadside to the array. The frequency of operation will be 491 MHz which is the frequency of TV channel 17.

To start the program type:

DIPARA [File_Name]

at the DOS command prompt. **File_Name** is the name of a file to receive the programs output summarizing the design. **File_Name** is optional, the program will prompt for a file name if it is missing. It can be any valid DOS file name, including the reserved names **PRN** (output will go to the standard printer), **CON** (output will go to the console), or **NUL** (output is discarded). As with any alphanumeric input to the program, case is not important. The program will accept either upper or lower case.

Program Input

Units for Linear Dimensions

The first request the program makes is for the units the user wants to use for linear dimensions. The prompt is shown in Figure 1. The program can handle units of **inches**, **feet**, **centimeters**, or **meters**. As described in the prompt, one of these is selected by typing **0**, **1**, **2**, or **3**, respectively. The program will only accept these four entries. If anything else is input an error message will be displayed and the user will be asked to try again. As shown in Figure 1, for this example **inches** will be used.

Source Frequency

Next the program requests the source frequency. This should be

```
Please specify the units you want to use for the array size and element
position from one of the following:

                    0 - inches,
                    1 - feet,
                    2 - centimeters,
                    3 - meters.

Make your selection by entering the number corresponding to your choice.

I want to use: 0
```

Figure 1 Prompt for units

```
Enter the desired source frequency (in GHz): .491
```

Figure 2 Prompt for Source Frequency

specified in GHz, as indicated in the prompt shown in Figure 2. For this example the desired source frequency is .491 GHz (491 MHz).

Beam Locations

The program next asks for the number of beams and their locations as shown in Figure 3.

```
When prompted enter the desired number of beams, and the AZIMUTH and ELEVATION
angle (in degrees) for each beam.

    Enter desired number of beams (1-3): 1
    Enter desired position of beam #1 (AZ, EL): 0. 0.
```

Figure 3 Prompt for number of beams and their locations

As indicated in the prompt for the number of beams, at least one beam is required and no more than three is allowed. If the number of beams is not in this range an error message is displayed and the user is asked to try again.

The beam locations are entered as azimuth and elevation angles in degrees. 0° azimuth, 0 ° elevation is perpendicular to the array panel, assuming no panel tilt (see below). Positive azimuth is clockwise (to your right) when looking in the direction of the beam. A positive elevation angle is up. These definitions correspond to those used in surveying. Although, there are no traps in the program to prevent you from doing so, it does not make sense to specify an azimuth or elevation angle outside the range ±90°. The program will not allow overlapping main beam directions and will display an error message if this is attempted.

When entering the position of each beam the azimuth angle should be entered first, then the elevation angle. The user is reminded of this by the **AZ, EL** symbols shown in parenthesis in the prompt. The numbers may be entered both on the same line (without an intervening carriage return) or on seperate lines. If the entries are on the same line seperate them with a space or a comma. If a carriage return does seperate the entries no additional prompts, indicating that an incomplete beam position has been entered, will be output by the program.

As shown in the figure, the beam is to be pointed perpendicular to the panel for the example design.

Panel Tilt

The array panel can be tilted in elevation. As shown in Figure 4,

```
The panel containing the array may be tilted in elevation.  At the prompt
specify the desired panel tilt by entering the elevation angle of the panel
normal.

     Enter desired panel tilt (in degrees): 0.
```

Figure 4 Prompt for Panel Tilt

the program next asks for the desired panel tilt. This is specified by entering the elevation angle of the panel **normal** in degrees. As with the beam elevation angle, a positive panel tilt is up. A zero degree panel tilt is chosen for the example design. The array panel is, thus, vertical.

Dipole Polarization

The polarization of the array is the next parameter requested as shown in Figure 5.

```
The dipoles making up the array can be tilted in the aperture.  At the prompt
enter the desired dipole tilt angle.  The angle should be specified in degrees
measured from the vertical.  The dipoles will be vertical for a 0 degree tilt
angle and horizontal for a 90 degree tilt angle.  A positive tilt angle is
assumed to be in a COUNTER-CLOCKWISE direction when looking in the same
direction as the array.

     Enter the desired dipole tilt (polarity): 90.
```

Figure 5 Prompt for array polarization

The polarization is specified by entering the dipole tilt angle in degrees. This tilt angle is in the plane of the panel. As indicated in the figure, the tilt angle is measured from vertical. That is, the dipoles will be vertical for a dipole tilt of 0° and horizontal for a dipole tilt of 90°. Both positive and negative tilt angles are allowed. A positive tilt angle is assumed to be in a **counter-clockwise** direction when looking in the same direction as the array. For the example design, the tilt angle is specified as 90°; the dipoles will be horizontal.

The Filter Network

To allow for removal of signals which are off frequency, a filter network can be placed on the array panel. The desired location of

```
Space must be left for the filter network along the

                (T)op edge, or
                (B)ottom edge

of the array panel. Indicate your choice by typing the corresponding letter
shown in parenthesis. I want: B
```

Figure 6 Prompt for Location of Filter Network

the filter network is asked for next, as shown in Figure 6. The prompt for the filter network location is dependent upon the requested dipole tilt angle. For

$$45^\circ \leq \mathbf{DipTilt} < 90^\circ$$

the prompt is as shown in the figure. For

$$0^\circ < \mathbf{DipTilt} < 45^\circ$$

(L)eft edge and **(R)ight edge** would replace **(T)op edge** and **(B)ottom edge** respectively. Here the angle supplied by the user has been adjusted so that **DipTilt** is between 0° and 90°.

The desired location is chosen by entering the letter shown in the parenthesis as indicated in the prompt. More than one letter may be typed but only the first letter is used. The program only accepts **T**, **B**, **L**, or **R**, ignoring case. Any other letters will result in an error message and a request to try again.

The filter network is to be placed along the bottom edge for the example design.

Ground Plane Information

The program next asks if a ground plane is desired, as shown in Figure 7.

```
Do you want to use a ground plane (y or n)? Y

    Please enter the dielectric constant of the SPACER material
    between the array substrate and ground plane
    (typically 1.033 for polystyrene): 1.033
```

Figure 7 Prompt for Ground Plane Information

As indicated in the prompt, enter either **Y** or **N** (again, case is ignored) in response to the prompt.

Responding with **Y** indicates that a ground plane is desired. The program will then ask for the dielectric constant of the material between the array panel and the ground plane. This value is used to compute the physical distance to the ground plane needed to build the array. This distance is recorded in the output file. Typical dielectric constants are less than 2.0 but no validation of the entered value is performed. For the user's convenience, the program lists the measured dielectric constant for polystryene, a common spacer material.

For the example design polystyrene was chosen for the spacer material.

If a ground plane is not required the spacer dielectric constant is not requested.

Panel Size

The program next asks for the desired size of the panel as shown in Figure 8.

```
At the prompt enter the desired dimensions for the array panel, including the
filter network. The array will be centered within this area after leaving
space for the filter network.

    Enter desired HORIZONTAL dimension (in in.): 65.
    Enter desired VERTICAL dimension (in in.): 65.

The actual array size is      65.000000 by      65.000000
The active array size is      65.000000 by      52.980800
There are  3 elements HORIZONTALLY and  3 elements VERTICALLY
  HORIZONTAL spacing=     16.025600 VERTICAL spacing=      16.025600
Is this OK (y or n)? Y
```

Figure 8 Prompt for Panel Size

The panel size should include the space for the filter network. The area containing the dipoles will be centered within the specified area after leaving space for the filter network.

The horizontal dimension is entered first and then the vertical dimension. The prompt includes the units requested earlier in the session. If the filter network is to be placed along the top or bottom edge the program checks the vertical dimension to insure enough space is present for the network. If not, an error message is displayed with a request to try again. A similar check is made when the filter network is to be along the left or right edge.

For the example design the array is to be 65 in. by 65 in. square. The horizontal and vertical dimensions, however, need not be the same.

Using the requested dimensions, the program computes the number and spacing of the dipoles required to point the beams in the previously requested directions and avoid grating lobes. As shown in the figure, the program displays the results of these computations and waits for confirmation from the user that everything is okay. If the number or spacing of the dipoles is not to the user's liking, answering **N** will allow the panel size to be changed.

Currently, the largest number of dipoles allowed horizontally or vertically is 20. If the panel size and beam angles are such that the computed number of dipoles exceeds 20 the program will issue an error message and allow the user to decrease the size of the panel.

Substrate Parameters

The final set of parameters requested by the program are the dielectric constant and thickness of the substrate material. The program prompt is shown in Figure 9.

```
In order to design the filter network the dielectric constant and thickness
of the SUBSTRATE must be specified. Please enter these values at the prompts.
(Teflon glass= 2.76  Mylar=2.25)

    Enter substrate DIELECTRIC CONSTANT: 2.25
    Enter substrate THICKNESS (in in.): 0.005
```

Figure 9 Prompt for Substrate Parameters

These parameters are required for the design of the matching and filter networks. The program lists the dielectric constant for Teflon glass and for Mylar, two common substrate materials. Also, the program, again, includes the required units in the prompt for the substrate thickness.

For the example design the substrate has a dielectric constant of 2.25 and is 5 mils thick. These values were measured from the mylar drafting film used in building this antenna.

Input Check

Before proceeding with the design of the array, the program next displays a summary showing the values of the previously entered parameters to be used in the design. As shown in Figure 10, the

```
The parameters which will be used in the design of the array are as follows:

     Source Frequency ...................... 0.491 GHz
     Position of Beam #1 (Az El) ........... 0  0 degrees
     Panel Tilt ............................ 0 degrees
     Dipole Tilt ........................... 90 degrees
     Location of the Filter Network ........ Bottom
     Gnd Plane Spacer Dielectric Constant .. 1.03300000
     Panel horizontal dimension ............ 65 in.
     Panel vertical dimension .............. 65 in.
     Substrate Dielectric Constant ......... 2.25
     Substrate thickness ................... 0.005 in.

Are these parameters ok (y or n)? Y
```

Figure 10 Prompt for Input Parameter Verification

user is then asked to verify that these values are correct. As indicated in the prompt, enter either **Y** or **N** (again, case is ignored) in response to the prompt.

Responding with **N** indicates that one or more of the parameters was incorrect. The program will return to the Figure 1 prompt and proceed to ask for each of the input parameters again. Answering **Y** to the Figure 10 prompt will cause the program to proceed with the design.

Program Output

The program now has all the information required to compute the driving point currents, the driving point impedances, and the principal plane patterns.

Computation of Pattern Cuts

The program is now ready to compute and display cuts of the radiation pattern. Since this is not always desired, the program asks the user if pattern plots are desired as shown in Figure 11.

```
Do you want to see the pattern plots (y or n)? Y
```

Figure 11 Prompt for Pattern Plots

As indicated in the prompt, the user should respond with either **Y**, to see the patterns, or **N**, to skip the pattern computation. Any other response will result in an error message and a request to try again.
If the patterns are desired the user is presented with a menu of allowed hard-copy devices as shown in Figure 12.

```
Graphics can be displayed on the following devices:

                    0 - an EGA monochrome display,
                    1 - an IBM CGA display,
                    2 - an HP 7470A plotter, or
                    3 - an HP LaserJet printer.

Please enter the number corresponding to your choice: 0
```

Figure 12 Prompt for Hard Copy device

The desired output device is chosen by entering the corresponding number. If something other than these numbers are entered an error message is displayed and the user is asked to try again. No checking for the existence of the selected device is performed so the user should be sure that the selected device is present.

The program expects the LaserJet printer to be connected as **PRN** or **LPT1** (they are equivalent). If the LaserJet is connected to the serial port the appropriate DOS **MODE** commands should have been executed before running the program.

The program expects the HP 7470A plotter to be connected to **COM1** and setup to run at 9600 bps, no parity, 8 data bits, 1 stop bit. The **Y/D** switch should be set to **D.** No DOS **MODE** command is necessary to initialize the **COM1** port.

If the CGA is selected as the hardcopy device the graphics will be displayed in the CGA monochrome graphics mode even if a color monitor is being used. The PLOT88 package being used to provide graphics only supports the mononchrome mode in graphics since it provides the highest resolution. The color graphics modes are not high enough in resolution to be of much use for this application anyway. Some graphics cards allow running CGA modes on a monochrome monitor. In order to display graphics on the monitor it may be necessary to put the card in CGA mode, using the DOS command, **MODE CO80**, before running the program.

Once the hard-copy device is selected, the user is prompted for the elevation angle desired for the azimuth cut, and the azimuth angle desired for the elevation cut, as shown in Figure 13.

```
Beginning AZIMUTH pattern plot.  Please enter the ELEVATION angle desired
for the cut (ELBeam=0): 0.

    Please wait - Computing the AZIMUTH pattern...

Beginning ELEVATION pattern plot.  Please enter the AZIMUTH angle desired
for the cut (AZBeam=0): 0.

    Please wait - Computing the ELEVATION pattern...
```

Figure 13 Prompt for Constant Angles for Patterns

For the user's convenience, the prompts include the beam angles, requested earlier, in parenthesis. Since the computation of the patterns takes some time, the program displays the **Please wait** messages while computing each cut. Also, the wording of these messages changes as the computations progress as an added assurance that the program is running and all is well. The program beeps the speaker when it is ready for more input.

After computation of the patterns, the program will display a message either requesting the user strike a key, or informing the user to wait while the hard-copy is made. The exact wording of the message is dependant upon the particular hard-copy device selected.

In the case of either the EGA or CGA displays the user should strike a key (the space bar is a good choice) to display the azimuth pattern plot. After completing the plot, the program will wait for the user to finish studying the pattern. When ready strike the space bar (or any other key) to display the elevation pattern. When the user is finished studying the elevation pattern again strike the space bar.

If the HP 7470A plotter is selected as the hard-copy device the program will prompt the user to load paper and strike any key to begin plotting. When the plotter is finished the program will stop and, again, wait for a key press to continue.

If the HP LaserJet printer is selected as the hard-copy device the program will display a **Please wait** message indicating that data is being sent to the printer. The printer **READY** light should be blinking during the transfer. The paper is automatically ejected from the printer when the plot is complete.

Once the hard-copy plots have been completed the program asks the user if another cut is desired, as shown in Figure 14. Responding with **Y** the user is again presented with the menu in Figure 12.

```
Do you want to see another cut (y or n)? N
```

Figure 14 Prompt for Another Cut

This allows the user to select either the laser printer or the plotter to get a paper copy of the patterns previously displayed on an EGA or CGA display, for example. The program then, again, asks for the elevation and azimuth angles for the cuts, recomputes the patterns, and displays them on the selected output device. When the program receives a **N** to the prompt in Figure 14 it begins the computation of the driving point impedances.

The patterns for the example design are shown in Figure 15. These patterns were produced on the HP LaserJet printer. Similar results are obtained when using the HP 7470A plotter.

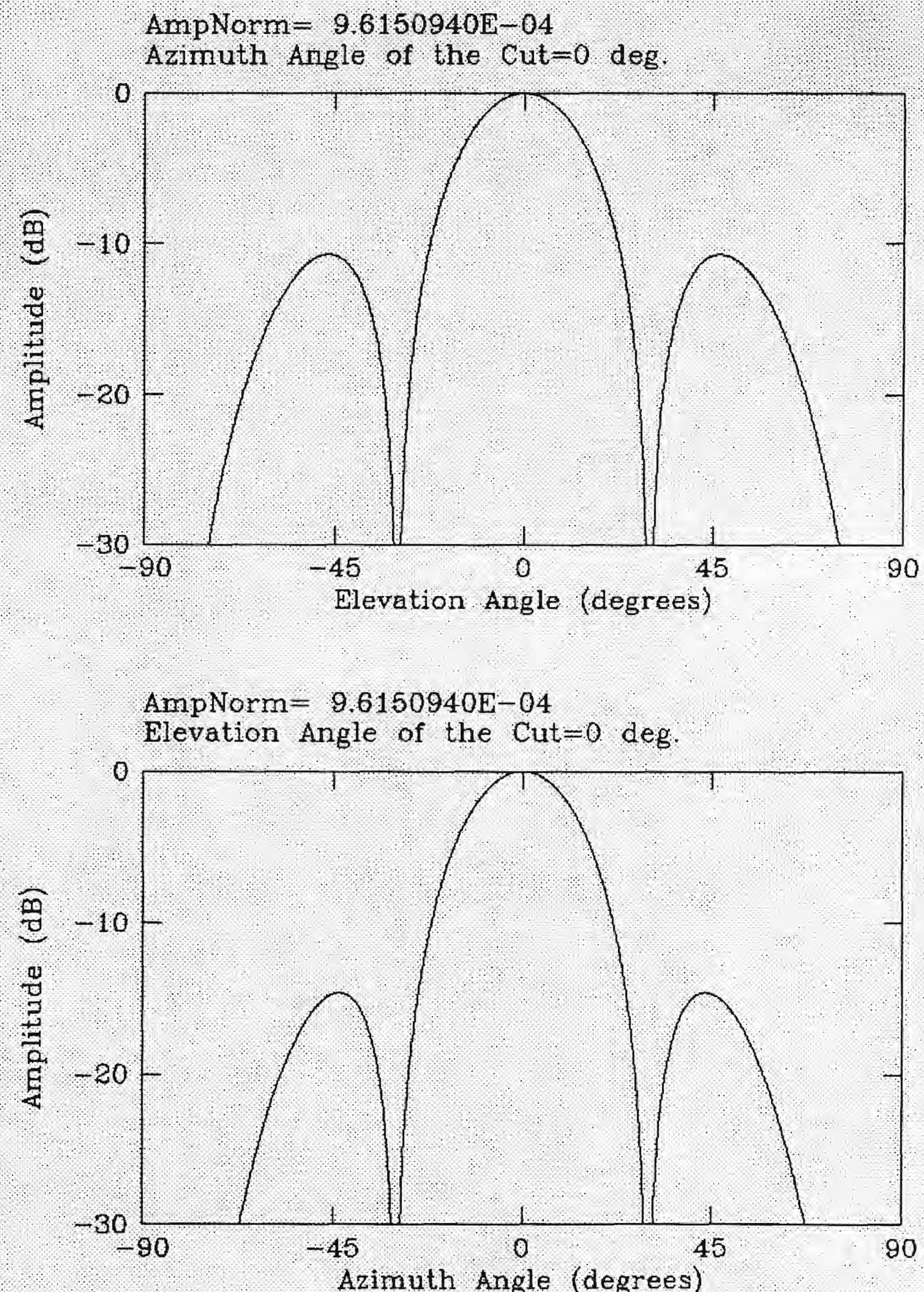

Figure 15 Pattern Cuts for Example Design

Computation of Driving Point Impedances

Depending on the size of the array, the computation of the driving point impedances may take some time. The example design, with 3 elements by 3 elements, takes approximately 20 seconds for this computation on an IBM XT. To reassure the user that the machine has not locked up, the program displays a **Please wait** while it computes the impedances. The speaker is sounded when the computation has finished.

Summary of Array Design

At this point, the program is ready to write a summary of the design to a file. If **File_Name** was specified on the DOS command line the program will write the data to **File_Name**. If **File_Name** was not specified the program will prompt for a file name as shown in Figure 16.

```
Enter name of file to receive design summary: PRN
```

Figure 16 Prompt for Output Filename

Any valid DOS file name is an acceptable answer to this prompt, including the DOS reserved names **PRN, CON,** or **NUL.**

If a disk file is specified as the output file, either on the command line or in answer to the Figure 16 prompt, the program checks to see if the file already exists. If it does the program will warn the user and ask if it is okay to overwrite the file. If the user indicates that the file should **not** be overwritten the program will ask for a new filename.

If the summary data is sent to a disk file the program displays a message to remind the user of the name of the file containing the data. This is especially helpful when the file name is entered on the command line.

The summary data for the example design is shown in Figure 17.

```
DIPOLE ARRAY DESIGN              9/14/1989  14:48:50
                *****
The operating frequency is   4.910000E-01 GHz  (Lambda=      24.038400 in.)
The beam is pointed at    0.000000E+00 deg. AZIMUTH   0.000000E+00 deg. ELEVATION
                *****
The array panel is tilted    0.000000E+00 degrees in ELEVATION
The array is         65.000000 in. by        65.000000 in.
There are              3 elements in the HORIZONTAL direction
                       3 elements in the VERTICAL direction
The dipole elements are spaced        16.025600 in. apart HORIZONTALLY
                                      16.025600 in. apart VERTICALLY
Each dipole is        11.177860 in. long and    2.500000E-01 in. wide
Each dipole is tilted       90.000000 degrees from VERTICAL
                *****
The filter network is located along the BOTTOM edge of the substrate
The substrate dielectric constant is          2.250000
The substrate is     5.000000E-03 in. thick
                *****
A ground plane is to be used        5.912830 in. from the array panel
The dielectric constant of the spacer material is        1.033000
                *****

     Array     |       Location         |               Driving Point
  Column   Row |Horizontal    Vertical  |      Current              Impedance
       1 |   1 |  16.5      |  22.5     |  1.00   /_  .000    |  87.4    /_ -7.09
       2 |   1 |  32.5      |  22.5     |  1.00   /_  .000    |  101.    /_ -14.3
       3 |   1 |  48.5      |  22.5     |  1.00   /_  .000    |  87.4    /_ -7.09
       1 |   2 |  16.5      |  38.5     |  1.00   /_  .000    |  65.1    /_ -24.4
       2 |   2 |  32.5      |  38.5     |  1.00   /_  .000    |  68.8    /_ -37.1
       3 |   2 |  48.5      |  38.5     |  1.00   /_  .000    |  65.1    /_ -24.4
       1 |   3 |  16.5      |  54.5     |  1.00   /_  .000    |  87.4    /_ -7.09
       2 |   3 |  32.5      |  54.5     |  1.00   /_  .000    |  101.    /_ -14.3
       3 |   3 |  48.5      |  54.5     |  1.00   /_  .000    |  87.4    /_ -7.09
               *****
Azimuth  Elevation  Gain dBi
  .000      .000     17.350
               *****
```

Figure 17 Summary Data for Example Design

Layout Data

To interface to the layout programs **DIPARA** next writes data required by these programs to a disk file on the current drive. The data is written to the file **LAYOUT.DAT**. Any old data which might exist in that file is overwritten.

Display of Array Schematic

Finally, the program can plot a schematic of the array. This is useful for checking the locations of the dipoles and that the tilt angle was entered correctly. The program asks the user if the display schematic is desired, as shown in Figure 18.

```
Do you want to see the array schematic (y or n)? Y
```

Figure 18 Prompt for Array Schematic

Answering **Y** to the prompt results in the Figure 12 prompt for selection of the graphics hard-copy device. The details of using this menu were described in the section on plotting the pattern cuts and will not be repeated here.

After the user chooses a hard-copy device the program displays a **Please wait** message while it makes the computations to plot the array schematic. When it is ready to display the schematic it will sound the speaker, display a message, and wait for the user to strike a key if the EGA, CGA or HP 7470A plotter was chosen as the hard-copy device. Just a message will be displayed if the laser printer is the hard-copy device.

Once the hard-copy has been finished the program will terminate and return to the DOS command line. It may be necessary to strike a key first, in which case a message to that effect will be displayed.

The array schematic for the example design is shown in Figure 19. It was produced on the laser printer but similar results will be obtained using the HP plotter.

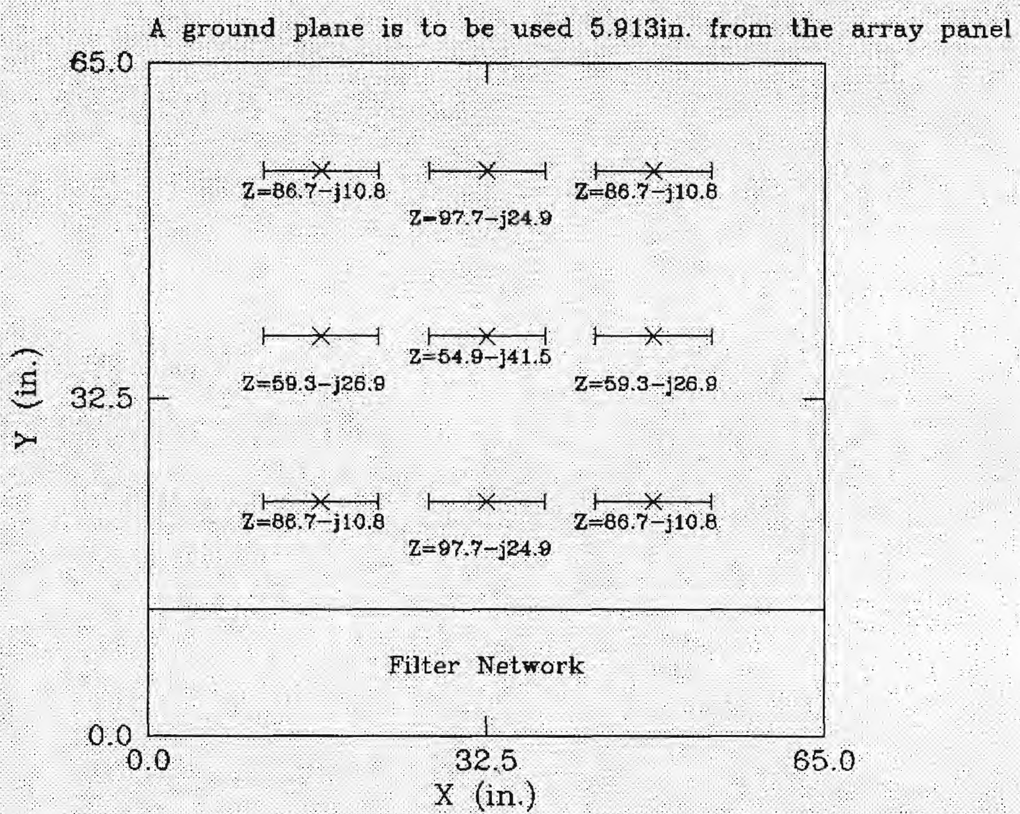

Figure 19 Array Schematic for the Example Design

E-21-F-21

Draft
Computer Operator
Manual (AA002)

Introduction

DIPARA is a strip dipole array design program. The user inputs the source frequency, main beam directions, array panel tilt, dipole tilt, location of a filter network, ground plane information, size of the array panel, and panel substrate dielectric constant and thickness. From this information the program determines the number of dipoles, their locations, and the driving point currents required to produce the desired beam direction. Also, the driving point impedances are determined for use in design of the matching and filter networks.

This user's manual describes how to run the program. To this end the reader is taken step by step through an example design.

Running the Program: an example design

The example design is for a single beam antenna with the beam broadside to the array. The frequency of operation will be 491 MHz which is the frequency of TV channel 17.

To start the program type:

DIPARA [File_Name]

at the DOS command prompt. **File_Name** is the name of a file to receive the programs output summarizing the design. **File_Name** is optional, the program will prompt for a file name if it is missing. It can be any valid DOS file name, including the reserved names **PRN** (output will go to the standard printer), **CON** (output will go to the console), or **NUL** (output is discarded). As with any alphanumeric input to the program, case is not important. The program will accept either upper or lower case.

Program Input

Units for Linear Dimensions

The first request the program makes is for the units the user wants to use for linear dimensions. The prompt is shown in Figure 1. The program can handle units of **inches**, **feet**, **centimeters**, or **meters**. As described in the prompt, one of these is selected by typing **0**, **1**, **2**, or **3**, respectively. The program will only accept these four entries. If anything else is input an error message will be displayed and the user will be asked to try again. As shown in Figure 1, for this example **inches** will be used.

Source Frequency

Next the program requests the source frequency. This should be

```
Please specify the units you want to use for the array size and element
position from one of the following:

                0 - inches,
                1 - feet,
                2 - centimeters,
                3 - meters.

Make your selection by entering the number corresponding to your choice.

I want to use: 0
```

Figure 1 Prompt for units

```
Enter the desired source frequency (in GHz): .491
```

Figure 2 Prompt for Source Frequency

specified in GHz, as indicated in the prompt shown in Figure 2. For this example the desired source frequency is .491 GHz (491 MHz).

Beam Locations

The program next asks for the number of beams and their locations as shown in Figure 3.

```
When prompted enter the desired number of beams, and the AZIMUTH and ELEVATION
angle (in degrees) for each beam.

   Enter desired number of beams (1-3): 1
   Enter desired position of beam #1 (AZ, EL): 0. 0.
```

Figure 3 Prompt for number of beams and their locations

As indicated in the prompt for the number of beams, at least one beam is required and no more than three is allowed. If the number of beams is not in this range an error message is displayed and the user is asked to try again.

The beam locations are entered as azimuth and elevation angles in degrees. 0° azimuth, 0 ° elevation is perpendicular to the array panel, assuming no panel tilt (see below). Positive azimuth is clockwise (to your right) when looking in the direction of the beam. A positive elevation angle is up. These definitions correspond to those used in surveying. Although, there are no traps in the program to prevent you from doing so, it does not make sense to specify an azimuth or elevation angle outside the range ±90°. The program will not allow overlapping main beam directions and will display an error message if this is attempted.

When entering the position of each beam the azimuth angle should be entered first, then the elevation angle. The user is reminded of this by the **AZ, EL** symbols shown in parenthesis in the prompt. The numbers may be entered both on the same line (without an intervening carriage return) or on seperate lines. If the entries are on the same line seperate them with a space or a comma. If a carriage return does seperate the entries no additional prompts, indicating that an incomplete beam position has been entered, will be output by the program.

As shown in the figure, the beam is to be pointed perpendicular to the panel for the example design.

Panel Tilt

The array panel can be tilted in elevation. As shown in Figure 4,

```
The panel containing the array may be tilted in elevation. At the prompt
specify the desired panel tilt by entering the elevation angle of the panel
normal.

    Enter desired panel tilt (in degrees): 0.
```

Figure 4 Prompt for Panel Tilt

the program next asks for the desired panel tilt. This is specified by entering the elevation angle of the panel **normal** in degrees. As with the beam elevation angle, a positive panel tilt is up. A zero degree panel tilt is chosen for the example design. The array panel is, thus, vertical.

Dipole Polarization

The polarization of the array is the next parameter requested as shown in Figure 5.

```
The dipoles making up the array can be tilted in the aperture. At the prompt
enter the desired dipole tilt angle. The angle should be specified in degrees
measured from the vertical. The dipoles will be vertical for a 0 degree tilt
angle and horizontal for a 90 degree tilt angle. A positive tilt angle is
assumed to be in a COUNTER-CLOCKWISE direction when looking in the same
direction as the array.

    Enter the desired dipole tilt (polarity): 90.
```

Figure 5 Prompt for array polarization

The polarization is specified by entering the dipole tilt angle in degrees. This tilt angle is in the plane of the panel. As indicated in the figure, the tilt angle is measured from vertical. That is, the dipoles will be vertical for a dipole tilt of 0° and horizontal for a dipole tilt of 90°. Both positive and negative tilt angles are allowed. A positive tilt angle is assumed to be in a **counter-clockwise** direction when looking in the same direction as the array. For the example design, the tilt angle is specified as 90°; the dipoles will be horizontal.

The Filter Network

To allow for removal of signals which are off frequency, a filter network can be placed on the array panel. The desired location of

```
Space must be left for the filter network along the

                (T)op edge, or
                (B)ottom edge

of the array panel. Indicate your choice by typing the corresponding letter
shown in parenthesis. I want: B
```

Figure 6 Prompt for Location of Filter Network

the filter network is asked for next, as shown in Figure 6. The prompt for the filter network location is dependent upon the requested dipole tilt angle. For

$$45^\circ \leq \textbf{DipTilt} < 90^\circ$$

the prompt is as shown in the figure. For

$$0^\circ < \textbf{DipTilt} < 45^\circ$$

(L)eft edge and **(R)ight edge** would replace **(T)op edge** and **(B)ottom edge** respectively. Here the angle supplied by the user has been adjusted so that **DipTilt** is between 0° and 90°.

The desired location is chosen by entering the letter shown in the parenthesis as indicated in the prompt. More than one letter may be typed but only the first letter is used. The program only accepts **T**, **B**, **L**, or **R**, ignoring case. Any other letters will result in an error message and a request to try again.

The filter network is to be placed along the bottom edge for the example design.

Ground Plane Information

The program next asks if a ground plane is desired, as shown in Figure 7.

```
Do you want to use a ground plane (y or n)? Y

     Please enter the dielectric constant of the SPACER material
     between the array substrate and ground plane
     (typically 1.033 for polystyrene): 1.033
```

Figure 7 Prompt for Ground Plane Information

As indicated in the prompt, enter either **Y** or **N** (again, case is ignored) in response to the prompt.

Responding with **Y** indicates that a ground plane is desired. The program will then ask for the dielectric constant of the material between the array panel and the ground plane. This value is used to compute the physical distance to the ground plane needed to build the array. This distance is recorded in the output file. Typical dielectric constants are less than 2.0 but no validation of the entered value is performed. For the user's convenience, the program lists the measured dielectric constant for polystryene, a common spacer material.

For the example design polystyrene was chosen for the spacer material.

If a ground plane is not required the spacer dielectric constant is not requested.

Panel Size

The program next asks for the desired size of the panel as shown in Figure 8.

```
At the prompt enter the desired dimensions for the array panel, including the
filter network.  The array will be centered within this area after leaving
space for the filter network.

     Enter desired HORIZONTAL dimension (in in.): 65.
     Enter desired VERTICAL dimension (in in.): 65.

The actual array size is       65.000000 by       65.000000
The active array size is       65.000000 by       52.980800
There are  3 elements HORIZONTALLY and   3 elements VERTICALLY
  HORIZONTAL spacing=      16.025600 VERTICAL spacing=       16.025600
Is this OK (y or n)? Y
```

Figure 8 Prompt for Panel Size

The panel size should include the space for the filter network. The area containing the dipoles will be centered within the specified area after leaving space for the filter network.

The horizontal dimension is entered first and then the vertical dimension. The prompt includes the units requested earlier in the session. If the filter network is to be placed along the top or bottom edge the program checks the vertical dimension to insure enough space is present for the network. If not, an error message is displayed with a request to try again. A similar check is made when the filter network is to be along the left or right edge.

For the example design the array is to be 65 in. by 65 in. square. The horizontal and vertical dimensions, however, need not be the same.

Using the requested dimensions, the program computes the number and spacing of the dipoles required to point the beams in the previously requested directions and avoid grating lobes. As shown in the figure, the program displays the results of these computations and waits for confirmation from the user that everything is okay. If the number or spacing of the dipoles is not to the user's liking, answering **N** will allow the panel size to be changed.

Currently, the largest number of dipoles allowed horizontally or vertically is 20. If the panel size and beam angles are such that the computed number of dipoles exceeds 20 the program will issue an error message and allow the user to decrease the size of the panel.

Substrate Parameters

The final set of parameters requested by the program are the dielectric constant and thickness of the substrate material. The program prompt is shown in Figure 9.

```
In order to design the filter network the dielectric constant and thickness
of the SUBSTRATE must be specified. Please enter these values at the prompts.
(Teflon glass= 2.76  Mylar=2.25)

   Enter substrate DIELECTRIC CONSTANT: 2.25
   Enter substrate THICKNESS (in in.): 0.005
```

Figure 9 Prompt for Substrate Parameters

These parameters are required for the design of the matching and filter networks. The program lists the dielectric constant for Teflon glass and for Mylar, two common substrate materials. Also, the program, again, includes the required units in the prompt for the substrate thickness.

For the example design the substrate has a dielectric constant of 2.25 and is 5 mils thick. These values were measured from the mylar drafting film used in building this antenna.

Input Check

Before proceeding with the design of the array, the program next displays a summary showing the values of the previously entered parameters to be used in the design. As shown in Figure 10, the

```
The parameters which will be used in the design of the array are as follows:

    Source Frequency ....................... 0.491 GHz
    Position of Beam #1 (Az El) ........... 0  0 degrees
    Panel Tilt ............................ 0 degrees
    Dipole Tilt ........................... 90 degrees
    Location of the Filter Network ........ Bottom
    Gnd Plane Spacer Dielectric Constant .. 1.03300000
    Panel horizontal dimension ............ 65 in.
    Panel vertical dimension .............. 65 in.
    Substrate Dielectric Constant ......... 2.25
    Substrate thickness ................... 0.005 in.

Are these parameters ok (y or n)? Y
```

Figure 10 Prompt for Input Parameter Verification

user is then asked to verify that these values are correct. As indicated in the prompt, enter either **Y** or **N** (again, case is ignored) in response to the prompt.

Responding with **N** indicates that one or more of the parameters was incorrect. The program will return to the Figure 1 prompt and proceed to ask for each of the input parameters again. Answering **Y** to the Figure 10 prompt will cause the program to proceed with the design.

Program Output

The program now has all the information required to compute the driving point currents, the driving point impedances, and the principal plane patterns.

Computation of Pattern Cuts

The program is now ready to compute and display cuts of the radiation pattern. Since this is not always desired, the program asks the user if pattern plots are desired as shown in Figure 11.

```
Do you want to see the pattern plots (y or n)? Y
```

Figure 11 Prompt for Pattern Plots

As indicated in the prompt, the user should respond with either **Y**, to see the patterns, or **N**, to skip the pattern computation. Any other response will result in an error message and a request to try again.

If the patterns are desired the user is presented with a menu of allowed hard-copy devices as shown in Figure 12.

```
Graphics can be displayed on the following devices:

                0 - an EGA monochrome display,
                1 - an IBM CGA display,
                2 - an HP 7470A plotter, or
                3 - an HP LaserJet printer.

Please enter the number corresponding to your choice: 0
```

Figure 12 Prompt for Hard Copy device

The desired output device is chosen by entering the corresponding number. If something other than these numbers are entered an error message is displayed and the user is asked to try again. No checking for the existence of the selected device is performed so the user should be sure that the selected device is present.

The program expects the LaserJet printer to be connected as **PRN** or **LPT1** (they are equivalent). If the LaserJet is connected to the serial port the appropriate DOS **MODE** commands should have been executed before running the program.

The program expects the HP 7470A plotter to be connected to **COM1** and setup to run at 9600 bps, no parity, 8 data bits, 1 stop bit. The **Y/D** switch should be set to **D**. No DOS **MODE** command is necessary to initialize the **COM1** port.

If the CGA is selected as the hardcopy device the graphics will be displayed in the CGA monochrome graphics mode even if a color monitor is being used. The PLOT88 package being used to provide graphics only supports the mononchrome mode in graphics since it provides the highest resolution. The color graphics modes are not high enough in resolution to be of much use for this application anyway. Some graphics cards allow running CGA modes on a monochrome monitor. In order to display graphics on the monitor it may be necessary to put the card in CGA mode, using the DOS command, **MODE CO80**, before running the program.

Once the hard-copy device is selected, the user is prompted for the elevation angle desired for the azimuth cut, and the azimuth angle desired for the elevation cut, as shown in Figure 13.

```
Beginning AZIMUTH pattern plot.  Please enter the ELEVATION angle desired
for the cut (ELBeam=0): 0.

    Please wait - Computing the AZIMUTH pattern...

Beginning ELEVATION pattern plot.  Please enter the AZIMUTH angle desired
for the cut (AZBeam=0): 0.

    Please wait - Computing the ELEVATION pattern...
```

Figure 13 Prompt for Constant Angles for Patterns

For the user's convenience, the prompts include the beam angles, requested earlier, in parenthesis. Since the computation of the patterns takes some time, the program displays the **Please wait** messages while computing each cut. Also, the wording of these messages changes as the computations progress as an added assurance that the program is running and all is well. The program beeps the speaker when it is ready for more input.

After computation of the patterns, the program will display a message either requesting the user strike a key, or informing the user to wait while the hard-copy is made. The exact wording of the message is dependant upon the particular hard-copy device selected.

In the case of either the EGA or CGA displays the user should strike a key (the space bar is a good choice) to display the azimuth pattern plot. After completing the plot, the program will wait for the user to finish studying the pattern. When ready strike the space bar (or any other key) to display the elevation pattern. When the user is finished studying the elevation pattern again strike the space bar.

If the HP 7470A plotter is selected as the hard-copy device the program will prompt the user to load paper and strike any key to begin plotting. When the plotter is finished the program will stop and, again, wait for a key press to continue.

If the HP LaserJet printer is selected as the hard-copy device the program will display a **Please wait** message indicating that data is being sent to the printer. The printer **READY** light should be blinking during the transfer. The paper is automatically ejected from the printer when the plot is complete.

Once the hard-copy plots have been completed the program asks the user if another cut is desired, as shown in Figure 14. Responding with **Y** the user is again presented with the menu in Figure 12.

```
Do you want to see another cut (y or n)? N
```

Figure 14 Prompt for Another Cut

This allows the user to select either the laser printer or the plotter to get a paper copy of the patterns previously displayed on an EGA or CGA display, for example. The program then, again, asks for the elevation and azimuth angles for the cuts, recomputes the patterns, and displays them on the selected output device. When the program receives a **N** to the prompt in Figure 14 it begins the computation of the driving point impedances.

The patterns for the example design are shown in Figure 15. These patterns were produced on the HP LaserJet printer. Similar results are obtained when using the HP 7470A plotter.

Computation of Driving Point Impedances

Depending on the size of the array, the computation of the driving point impedances may take some time. The example design, with 3 elements by 3 elements, takes approximately 20 seconds for this computation on an IBM XT. To reassure the user that the machine has not locked up, the program displays a **Please wait** while it computes the impedances. The speaker is sounded when the computation has finished.

Summary of Array Design

At this point, the program is ready to write a summary of the design to a file. If **File_Name** was specified on the DOS command line the program will write the data to **File_Name.** If **File_Name** was not specified the program will prompt for a file name as shown in Figure 16.

```
Enter name of file to receive design summary: PRN
```

Figure 16 Prompt for Output Filename

Any valid DOS file name is an acceptable answer to this prompt, including the DOS reserved names **PRN, CON,** or **NUL.**

If a disk file is specified as the output file, either on the command line or in answer to the Figure 16 prompt, the program checks to see if the file already exists. If it does the program will warn the user and ask if it is okay to overwrite the file. If the user indicates that the file should **not** be overwritten the program will ask for a new filename.

If the summary data is sent to a disk file the program displays a message to remind the user of the name of the file containing the data. This is especially helpful when the file name is entered on the command line.

The summary data for the example design is shown in Figure 17.

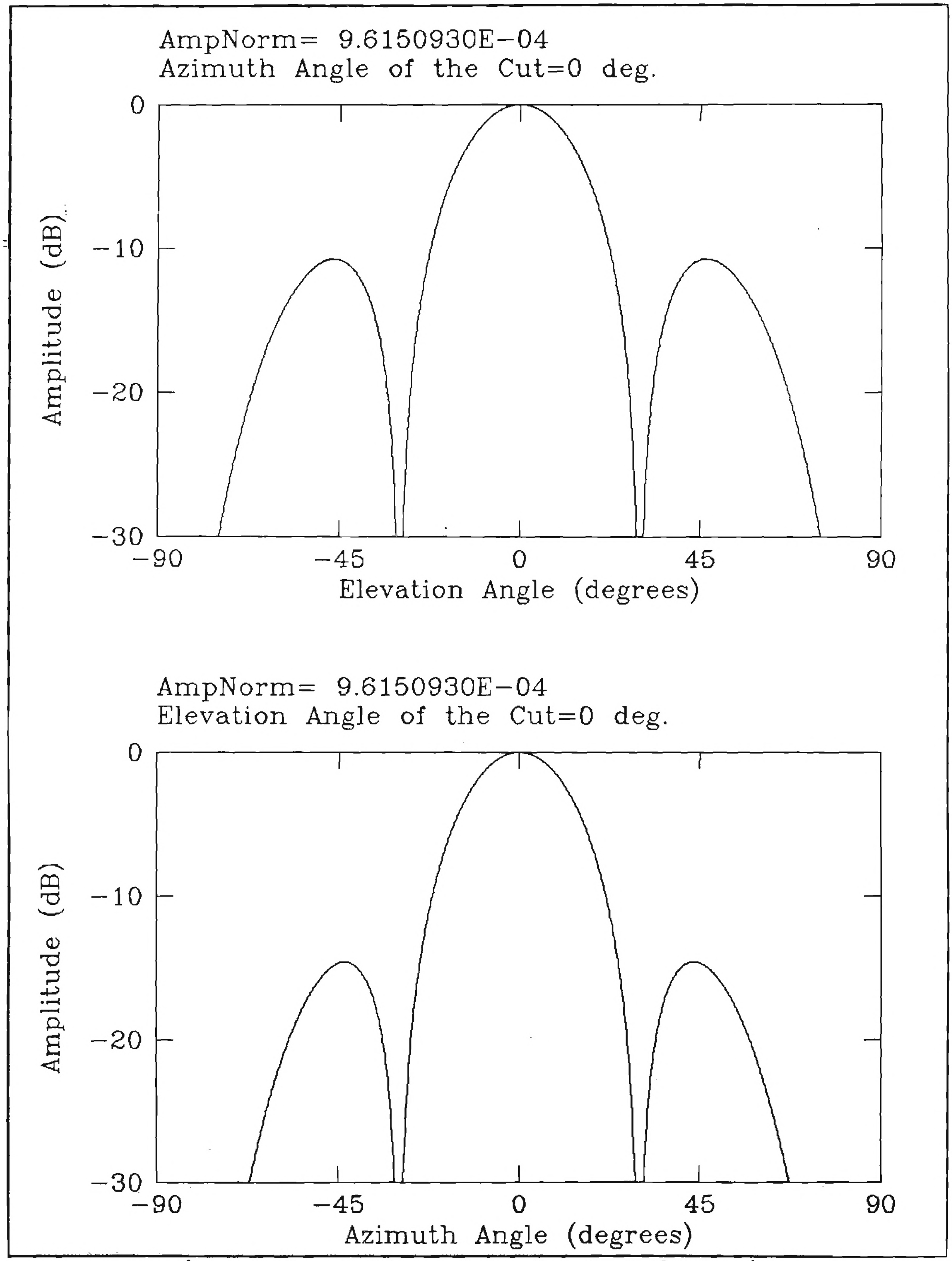

Figure 15 Pattern Cuts for Example Design

DIPOLE ARRAY DESIGN 9/14/1989 14:48:50

*****

The operating frequency is 4.910000E-01 GHz (Lambda= 24.038400 in.)
The beam is pointed at 0.000000E+00 deg. AZIMUTH 0.000000E+00 deg. ELEVATION

*****

The array panel is tilted 0.000000E+00 degrees in ELEVATION
The array is 65.000000 in. by 65.000000 in.
There are 3 elements in the HORIZONTAL direction
3 elements in the VERTICAL direction
The dipole elements are spaced 16.025600 in. apart HORIZONTALLY
16.025600 in. apart VERTICALLY
Each dipole is 11.177860 in. long and 2.500000E-01 in. wide
Each dipole is tilted 90.000000 degrees from VERTICAL

*****

The filter network is located along the BOTTOM edge of the substrate
The substrate dielectric constant is 2.250000
The substrate is 5.000000E-03 in. thick

*****

A ground plane is to be used 5.912830 in. from the array panel
The dielectric constant of the spacer material is 1.033000

*****

Array		Location		Driving Point	
Column	Row	Horizontal	Vertical	Current	Impedance
1	1	16.5	22.5	1.00 /_ .000	87.4 /_ -7.09
2	1	32.5	22.5	1.00 /_ .000	101. /_ -14.3
3	1	48.5	22.5	1.00 /_ .000	87.4 /_ -7.09
1	2	16.5	38.5	1.00 /_ .000	65.1 /_ -24.4
2	2	32.5	38.5	1.00 /_ .000	68.8 /_ -37.1
3	2	48.5	38.5	1.00 /_ .000	65.1 /_ -24.4
1	3	16.5	54.5	1.00 /_ .000	87.4 /_ -7.09
2	3	32.5	54.5	1.00 /_ .000	101. /_ -14.3
3	3	48.5	54.5	1.00 /_ .000	87.4 /_ -7.09

*****

Azimuth	Elevation	Gain dBi
.000	.000	17.350

*****

Figure 17 Summary Data for Example Design

Layout Data

To interface to the layout programs **DIPARA** next writes data required by these programs to a disk file on the current drive. The data is written to the file **LAYOUT.DAT**. Any old data which might exist in that file is overwritten.

Display of Array Schematic

Finally, the program can plot a schematic of the array. This is useful for checking the locations of the dipoles and that the tilt angle was entered correctly. The program asks the user if the display schematic is desired, as shown in Figure 18.

```
Do you want to see the array schematic (y or n)? Y
```

Figure 18 Prompt for Array Schematic

Answering **Y** to the prompt results in the Figure 12 prompt for selection of the graphics hard-copy device. The details of using this menu were described in the section on plotting the pattern cuts and will not be repeated here.

After the user chooses a hard-copy device the program displays a **Please wait** message while it makes the computations to plot the array schematic. When it is ready to display the schematic it will sound the speaker, display a message, and wait for the user to strike a key if the EGA, CGA or HP 7470A plotter was chosen as the hard-copy device. Just a message will be displayed if the laser printer is the hard-copy device.

Once the hard-copy has been finished the program will terminate and return to the DOS command line. It may be necessary to strike a key first, in which case a message to that effect will be displayed.

The array schematic for the example design is shown in Figure 19. It was produced on the laser printer but similar results will be obtained using the HP plotter.

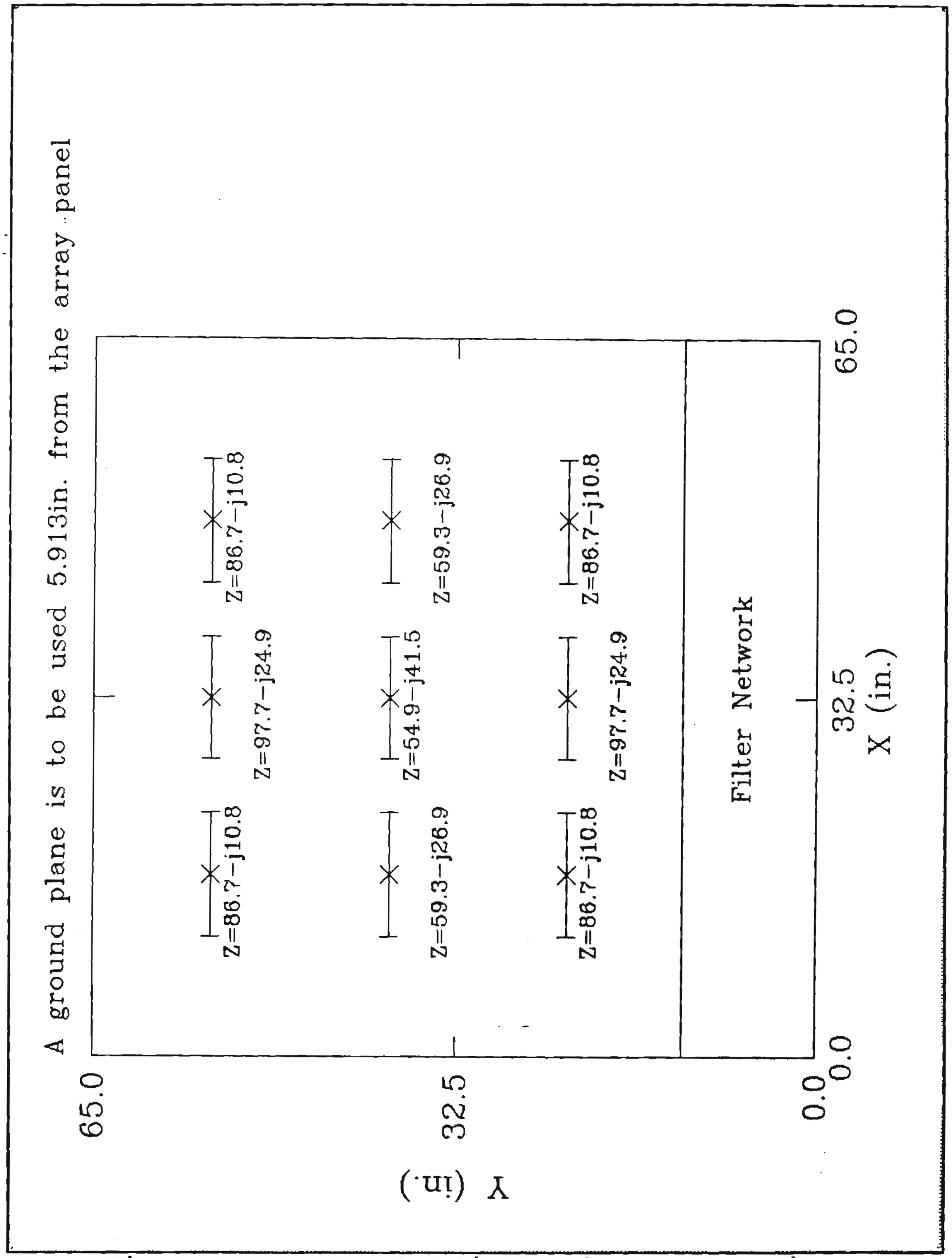

Figure 19 Array Schematic for the Example Design

E-21-F21

A PLANAR SURFACE, DIPOLE, PHASED ARRAY COMPUTER AIDED DESIGN AND LAYOUT PROGRAM

COMPUTER PROGRAM

Contract Number: MDA970-88-C-8145

Prepared by

Georgia Institute of Technology
School of Electrical Engineering
Atlanta, Georgia

A PLANAR SURFACE, DIPOLE, PHASED ARRAY COMPUTER AIDED DESIGN AND LAYOUT PROGRAM

COMPUTER PROGRAM

Contract Number: MDA970-88-C-8145

Prepared by

Georgia Institute of Technology
School of Electrical Engineering
Atlanta, Georgia

```
$PAGESIZE:80
$LINESIZE:132
$MESSAGE:'Compiling PROGRAM DIPARA'
$TITLE: 'PROGRAM DIPARA.FOR'
$PAGE
$NOTRUNCATE
      PROGRAM DIPARA
C
C   LAST MODIFIED  8/22/88.  BY R. WILSON.
C   *************  9/27/88.  BY R. WILSON.
C        1) Added call to "DRWARA" to plot a picture of the array.
C   ************* 11/01/88.  BY R. WILSON.
C        1) Changed from using "THBEAM" and "PHIBEAM" (spherical angles to
C           directly using "AZBEAM" and "ELBEAM".
C        2) Added call to "ARAPAT" to plot array azimuth and elevation
C           patterns.
C   ************* 12/11/88.  BY R. WILSON.
C        1) Added handling of dipole tilt (Tilt of Linear Polarization).
C   *************  3/09/89.  BY R. WILSON.
C        1) Added handling of multiple beams.
C   *************  4/04/89.  BY R. WILSON.
C        1) Added handling of ground plane.
C   *************  4/23/89.  BY R. WILSON.
C        1) Moved computation of "DRVVOLT" and "ZDRVIMP" into "DRVPNT".
C        2) This allows the program to work for large arrays on the PC since
C           only one row/column of the system impedance matrix is stored.
C   *************  5/08/89.  BY R. WILSON.
C        1) Removed computation of "DRVVOLT" since it is not used.
C   *************  5/11/89.  BY R. WILSON.
C        1) Added computation of gain at main beam locations.
C   *************  5/16/89.  BY R. WILSON.
C        1) Added compensation for non-air spacer material in the
C           computed value of the ground plane distance.
C        2) Two distances are now computed, "DGNDPLD", and "DGNDPLA".
C   *************  6/06/89.  BY R. WILSON.
C        1) Corrected method of loading impedances in "DRVPNT" to account
C           for the fact that when the dipoles are tilted (not 0 or 90 degrees)
C           the impednaces are functions of SIGN(IVOL-ICUR)*SIGN(JVOL-JCUR)
C           not ABS(IVOL-ICUR)*ABS(JVOL-JCUR).
C   *************  6/21/89.  BY R. WILSON.
C         1) Changed to using an array, "GNDPLN", to hold the two ground plane
C            distances and added the spacer dielectric constant as the third
C            value stored in "GNDPLN".
C   *************  8/08/89.  BY R. WILSON
C         1) Added call to OPNFILE to open the output file.
C         2) OPNFILE can get the name of the file to open from the
C            command line.  If it doesn't find the name on the command line
C            it prompts the user for the name.
C         3) This capability is provided by the fortran library,  however,
C            the prompt provided is not very flexible.  The prompt from
C            OPNFILE is programmer changeable.
C   *************  9/13/89.  BY R. WILSON.
C         1) Added call to SHOWIN to allow the user to check his inputs to
C            the program before the design is performed.
C         2) Moved writing the design summary from the main program to SUMMARY.
C
C   "DIPARA" designs a two dimensional dipole array.  The size of the array,
C   the frequency of operation, and the main beam direction are specified by
C   the user.
C
      PARAMETER (NXELMX=20,NYELMX=20,NXYELMX=NXELMX*NYELMX)
      PARAMETER (MXNBEAM=3)
      COMPLEX ZDRVIMP(NXELMX,NYELMX),ZSYSROW(NXYELMX,2)
      COMPLEX DRVCURR(NXELMX,NYELMX)
      COMPLEX REC2POL
      REAL XLOCATE(NXELMX),YLOCATE(NYELMX)
      REAL AZBEAM(MXNBEAM),ELBEAM(MXNBEAM),GAIN(MXNBEAM),DIPTILT(3)
      REAL GNDPLN(3),FREQ2KO,KO,LAMBDA
      LOGICAL YESNOTF,INPUTOK
```

```
      CHARACTER DATE*10,TIME*8,FNAME*32,UNITS*2,LFILNET*1
      CHARACTER DATSTMP*20
C
C   Initialize the date and time stamp for the output.
C
      DATSTMP=DATE()//'  '//TIME()
      INPUTOK=.FALSE.
      DO WHILE (.NOT. INPUTOK)
C
C   Obtain the units desired for length and position specifications.
C
         CALL GETUNIT(UNITS,FREQ2KO)
C
C   Obtain the operating frequency.
C
         CALL GETFREQ(FREQGHZ,FREQ2KO,KO,LAMBDA)
C
C   Obtain direction of beam.
C
         CALL GETBEAM(MXNBEAM,NUMBEAM,AZBEAM,ELBEAM)
C
C   Obtain the array geometry.
C
         CALL GETGEOM(UNITS,LAMBDA,NUMBEAM,AZBEAM,ELBEAM,PANTILT,
     $      DIPTILT,XSIZE,YSIZE,NXEL,NXELMX,NYEL,NYELMX,XLOCATE,YLOCATE,
     $      DIPWID,DIPLNG,DX,DY,LFILNET,SFILNET,GNDPLN)
C
C   Obtain substrate dielectric constant and thickness.
C
         CALL GSUBSTR(UNITS,ERSUB,THSUB)
C
C   Display the inputs for user approval.
C
         CALL SHOWIN(UNITS,FREQGHZ,NUMBEAM,AZBEAM,ELBEAM,PANTILT,
     $      DIPTILT(3),LFILNET,GNDPLN,XSIZE,YSIZE,ERSUB,THSUB,INPUTOK)
      END DO
C
C   Load driving point current array.
C
      CALL LDCURNT(KO,XLOCATE,YLOCATE,NXEL,NXELMX,NYEL,NUMBEAM,AZBEAM,
     $  ELBEAM,PANTILT,DRVCURR)
C
C   Plot the azimuth and elevation patterns.
C
      WRITE(*,*)
      IF(YesNoTF('Do you want to see the pattern plots'))THEN
         CALL ARAPAT(KO,NUMBEAM,AZBEAM,ELBEAM,PANTILT,DIPLNG,DIPTILT,
     $      GNDPLN(1),XLOCATE,YLOCATE,NXEL,NXELMX,NYEL,DRVCURR)
      ENDIF
C
C   Compute the driving point impedances.
C
      WRITE(*,100)
  100 FORMAT(/1X,'Please wait - computing the driving point',
     $  1X,'impedances...')
      CALL DRVPNT(LAMBDA,NXEL,NXELMX,NYEL,XLOCATE,YLOCATE,DIPWID,
     $  DIPLNG,DIPTILT,GNDPLN(1),DRVCURR,ZDRVIMP,ZSYSROW,NXYELMX)
      CALL BEEP
C
C   Compute the main beam gain.
C
      CALL ARAGAIN(KO,NUMBEAM,AZBEAM,ELBEAM,PANTILT,DIPLNG,DIPTILT,
     $  GNDPLN(1),XLOCATE,YLOCATE,NXEL,NXELMX,NYEL,DRVCURR,ZDRVIMP,GAIN)
C
C   Write the array parameters to a file.
C
      FNAME=' '
      CALL OPNFILE(1,'OUTPUT',FNAME,NAMLGTH,
     $  'Enter name of file to receive design summary:',1)
```

```
      CALL SUMMARY(DATSTMP,UNITS,FREQGHZ,LAMBDA,NUMBEAM,AZBEAM,ELBEAM,
     $  PANTILT,XSIZE,YSIZE,NXEL,NXELMX,NYEL,DX,DY,DIPLNG,DIPWID,
     $  DIPTILT,LFILNET,ERSUB,THSUB,GNDPLN,XLOCATE,YLOCATE,DRVCURR,
     $  ZDRVIMP,GAIN)
C
C
C   WRITE THE FIRST AND "NXEL"TH ROW OF THE SYSTEM IMPEDANCE MATRIX TO THE
C   FILE.   CLOSE THE OUTPUT FILE ON TAPE1.
C
c      NXYEL=NXEL*NYEL
c      DO 5 J=1,2,1
c         WRITE(1,*)'ZSYSROW(I,',J,')'
c         WRITE(1,*)(ZSYSROW(I,J),I=1,NXYEL,1)
c    5 CONTINUE
C
C   Close the output file on TAPE1.  Remind the user to which file the output
C   was written if the file is on disk.  Send form-feed if file was sent to the
C   printer (PRN or LPT1).  BIOPR is a PLOT88 routine.
C
      CLOSE(1)
      IF(FNAME.NE.'CON' .AND. FNAME.NE.'PRN' .AND. FNAME.NE.'NUL')
     $  WRITE(*,*)'Array data written to file ',FNAME
      IF(FNAME .EQ. 'PRN')CALL BIOPR(0,CHAR(12),1)
C
C   Write data to file for use by the layout programs.
C
      OPEN(1,FILE='LAYOUT.DAT')
      TOINCH=UNIT2IN(1.0,UNITS)
C
      WRITE(1,*)FREQGHZ
      WRITE(1,*)XSIZE*TOINCH,YSIZE*TOINCH
      WRITE(1,*)ERSUB,THSUB*TOINCH
      WRITE(1,*)DIPLNG*TOINCH,DIPWID*TOINCH,DIPTILT(3)
      WRITE(1,'(1X,A1,1X,1PG15.6)')LFILNET,SFILNET*TOINCH
      WRITE(1,*)NXEL,NYEL
      DO 7 J=1,NYEL,1
         Y=YLOCATE(J)*TOINCH
         DO 6 I=1,NXEL,1
            X=XLOCATE(I)*TOINCH
            WRITE(1,'(2(1X,1PG15.6),4(1X,1PG15.6))')X,Y,
     $          REC2POL(DRVCURR(I,J),.FALSE.),
     $          REC2POL(ZDRVIMP(I,J),.FALSE.)
    6    CONTINUE
    7 CONTINUE
      CLOSE(1)
C
C   Draw the dipole array.
C
      WRITE(*,*)
      IF(YesNoTF('Do you want to see the array schematic'))THEN
         CALL DRWARA(UNITS,XSIZE,YSIZE,NXEL,NYEL,DIPLNG,DIPTILT,LFILNET,
     $      SFILNET,GNDPLN(2),XLOCATE,YLOCATE,ZDRVIMP,NXELMX)
      ENDIF
C
      END
```

```fortran
$PAGESIZE:80
$LINESIZE:132
$MESSAGE:'Compiling ARAFACT'
$SUBTITLE: '    FUNCTION ARAFACT'
$PAGE
$NOTRUNCATE
      FUNCTION ARAFACT(KO,KDIR,DGNDPLN,XLOCATE,YLOCATE,DRVCURR,NXEL,
     $  NXELMX,NYEL)
C
C    LAST MODIFIED  1/26/89.  BY R. WILSON.
C    *************  3/12/89.  BY R. WILSON.
C         1)  ADDED HANDLING OF A GROUND PLANE.
C
C    "ARAFACT" COMPUTES THE ARRAY FACTOR FOR AN "NXEL" BY "NYEL" ARRAY OF
C    POINT SOURCES.  THE ARRAY MAY BE BACKED BY A GROUND PLANE.
C
C    "KO" IS THE FREE SPACE WAVENUMBER.
C
C    "KDIR" IS A UNIT VECTOR OF DIRECTION COSINES SPECIFYING THE DIRECTION
C    TO THE FIELD POINT IN QUESTION.
C
C    "DGNDPLN" IS THE DISTANCE TO THE GROUND PLANE.  THE UNITS FOR "DGNDPLN"
C    SHOULD BE COMPATIBLE WITH THOSE USED FOR "KO", I.E. KO*DGNDPLN SHOULD
C    HAVE UNITS OF RADIANS.  IF "DGNDPLN" .LE. 0.0 THEN "ARAFACT" ASSUMES THAT
C    THERE IS NO GROUND PLANE PRESENT.
C
C    "XLOCATE" AND "YLOCATE" ARE ARRAYS SPECIFYING THE X AND Y LOCATIONS,
C    RESPECTIVELY, OF THE POINT SOURCES.  THERE UNITS SHOULD BE COMPATIBLE WITH
C    THOSE USED FOR "KO", I.E. KO*XLOCATE(.) SHOULD HAVE UNITS OF RADIANS.
C
C    "DRVCURR" IS A TWO DIMENSIONAL ARRAY CONTAINING THE COMPLEX AMPLITUDES OF
C    EACH OF THE POINT SOURCES.
C
C    "NXEL" AND "NYEL" ARE THE NUMBER OF POINT SOURCES IN THE X AND Y DIRECTION,
C    RESPECTIVELY.  "NXELMX" IS THE ROW DIMENSION OF THE "DRVCURR" IN THE
C    CALLING ROUTINE.
C
      COMPLEX ARAFACT,DRVCURR(NXELMX,NYEL)
      REAL XLOCATE(NXEL),YLOCATE(NYEL)
      REAL KDIR(3),KODIR(2),KO,KXX,KYY
C
C    COMPUTE THE ARRAY FACTOR SUM.
C
      KODIR(1)=KO*KDIR(1)
      KODIR(2)=KO*KDIR(2)
      ARAFACT=CMPLX(0.0,0.0)
      DO 2 J=1,NYEL,1
         KYY=KODIR(2)*YLOCATE(J)
         DO 1 I=1,NXEL,1
            KXX=KODIR(1)*XLOCATE(I)
            ARAFACT=ARAFACT+DRVCURR(I,J)*CEXP(CMPLX(0.0,(KXX+KYY)))
    1    CONTINUE
    2 CONTINUE
C
C    INCLUDE FACTOR TO ACCOUNT FOR A GROUND PLANE.
C
      IF(DGNDPLN .GT. 0.0)THEN
         ARAFACT=(1.0-CEXP(CMPLX(0.0,-2.0*DGNDPLN*KO*KDIR(3))))*ARAFACT
      ENDIF
      RETURN
      END
```

```
$PAGESIZE:80
$LINESIZE:132
$MESSAGE:'Compiling ARAGAIN'
$SUBTITLE: '    SUBROUTINE ARAGAIN'
$PAGE
$NOTRUNCATE
      SUBROUTINE ARAGAIN(KO,NUMBEAM,AZBEAM,ELBEAM,PANTILT,DIPLNG,
     $  DIPTILT,DGNDPLN,XLOCATE,YLOCATE,NXEL,NXELMX,NYEL,DRVCURR,
     $  ZDRVIMP,GAIN)
C
C   LAST MODIFIED 5/11/89.  BY R. WILSON.
C
C   "ARAGAIN" COMPUTES THE GAIN FOR AN "NXEL" BY "NYEL" ARRAY OF DIPOLES. THE
C   THE ARRAY MAY HAVE MULTIPLE BEAMS AND BE BACKED BY A GROUND PLANE.
C
C   "KO" IS THE FREE SPACE WAVENUMBER OF THE SOURCE.
C
C   "NUMBEAM" IS THE NUMBER OF MAIN BEAMS GENERATED BY THE ARRAY.
C
C   "AZBEAM" AND "ELBEAM" ARE ARRAYS WHICH HOLD THE AZIMUTH AND ELEVATION
C   ANGLES, RESPECTIVELY, FOR EACH MAIN BEAM.  THE ANGLES ARE ASSUMED TO BE IN
C   DEGREES.
C
C   "PANTILT" IS THE ELEVATION ANGLE OF THE PANEL NORMAL IN DEGREES.  WHEN
C   "PANTILT" IS 0 THE PANEL IS PERPENDICULAR TO THE GROUND.
C
C   "DIPLNG" IS THE LENGTH OF EACH DIPOLE IN THE ARRAY.  IT ASSUMED TO BE IN
C   UNITS CONSISTENT WITH "KO".
C
C   "DIPTILT" IS AN ARRAY CONTAINING A UNIT DIRECTION VECTOR SPECIFYING THE
C   DIRECTION OF THE DIPOLES IN THE APERTURE PLANE.  IT SHOULD BE DIMENSIONED
C   THREE (3) IN THE CALLING PROGRAM.  SINCE THE Z COMPONENT OF "DIPTILT" WILL
C   ALWAYS BE 0 THE ANGLE OF THE DIPOLES, IN DEGREES, WITH RESPECT TO THE
C   APERTURE VERTICAL IS STORED IN "DIPTILT(3)".
C
C   "DGNDPLN" IS THE DISTANCE TO THE GROUND PLANE.  THE UNITS FOR "DGNDPLN"
C   SHOULD BE COMPATIBLE WITH THOSE USED FOR "KO", I.E. KO*DGNDPLN SHOULD
C   HAVE UNITS OF RADIANS.  IF "DGNDPLN" .LE. 0.0 THEN "ARAFACT" ASSUMES THAT
C   THERE IS NO GROUND PLANE PRESENT.
C
C   "XLOCATE" AND "YLOCATE" ARE ARRAYS SPECIFYING THE X AND Y LOCATIONS,
C   RESPECTIVELY, OF THE POINT SOURCES.  THERE UNITS SHOULD BE COMPATIBLE WITH
C   THOSE USED FOR "KO", I.E. KO*XLOCATE(.) SHOULD HAVE UNITS OF RADIANS.
C
C   "NXEL" AND "NYEL" ARE THE NUMBER OF POINT SOURCES IN THE X AND Y DIRECTION,
C   RESPECTIVELY.  "NXELMX" IS THE ROW DIMENSION OF THE "DRVCURR" IN THE
C   CALLING ROUTINE.
C
C   "DRVCURR" IS A TWO DIMENSIONAL ARRAY CONTAINING THE COMPLEX AMPLITUDES OF
C   EACH OF THE POINT SOURCES.
C
C   "ZDRVIMP" IS A TWO DIMENSIONAL COMPLEX ARRAY CONTAINING THE DRIVING POINT
C   IMPEDANCES FOR EACH OF THE DIPOLES IN THE ANTENNA.
C
      COMPLEX ARAFACT,DIPPAT,DRVCURR(NXELMX,NYEL),ZDRVIMP(NXELMX,NYEL)
      REAL XLOCATE(NXEL),YLOCATE(NYEL)
      REAL AZBEAM(NUMBEAM),ELBEAM(NUMBEAM),GAIN(NUMBEAM)
      REAL DIPTILT(3),KBEAM(3),KO
C
C   COMPUTE THE INPUT POWER.
C
      POWERIN=0.0
      DO 1 J=1,NYEL,1
         DO 1 I=1,NXEL,1
            POWERIN=POWERIN+REAL(ZDRVIMP(I,J))*CABS(DRVCURR(I,J))**2
    1 CONTINUE
C
C   COMPUTE THE GAIN.
C
```

```
      AMPNORM=DIPPATM(K0,DIPLNG)
      AMPNORM=120.0/AMPNORM**2
      DO 2 I=1,NUMBEAM,1
         CALL AZEL2K(AZBEAM(I),ELBEAM(I),PANTILT,KBEAM)
         GAIN(I)=AMPNORM*CABS(DIPPAT(KBEAM,DIPTILT)*
     $     ARAFACT(K0,KBEAM,DGNDPLN,XLOCATE,YLOCATE,DRVCURR,NXEL,
     $     NXELMX,NYEL))**2/POWERIN
         GAIN(I)=10.0*ALOG10(GAIN(I))
    2 CONTINUE
      RETURN
      END
```

```
$PAGESIZE:80
$LINESIZE:132
$MESSAGE:'Compiling ARAPAT'
$SUBTITLE: '    SUBROUTINE ARAPAT'
$PAGE
$NOTRUNCATE
      SUBROUTINE ARAPAT(KO,NUMBEAM,AZBEAM,ELBEAM,PANTILT,DIPLNG,DIPTILT,
     $  DGNDPLN,XLOCATE,YLOCATE,NXEL,NXELMX,NYEL,DRVCURR)
C
C   LAST MODIFIED 11/01/88.  BY R. WILSON.
C   *************  3/09/89.  BY R. WILSON.
C        1) Added handling of multiple beams.
C   *************  4/12/89.  BY R. WILSON.
C        1) Added handling of a ground plane.
C   *************  7/20/89.  BY R. WILSON.
C        1) Added trapping of operator data entry errors.
C        2) Previously a typing mistake would terminate the program with a
C           run-time error message and a return to DOS.
C
C   "ARAPAT" plots the array AZIMUTH and ELEVATION patterns.
C
      PARAMETER (NPTS=1001)
      COMPLEX DRVCURR(NXELMX,NYEL),ARAFACT,DIPPAT
      REAL XLOCATE(NXEL),YLOCATE(NYEL),AMP(NPTS),AZEL(NPTS)
      REAL AZBEAM(NUMBEAM),ELBEAM(NUMBEAM)
      REAL DIPTILT(3),BOXDIR(3),KDIR(3),KO
      INTEGER OFFSET
      LOGICAL YESNOTF
      CHARACTER AXISLBL(2)*80
      COMMON/LINAXC/MAXDIG
      COMMON/PLOT88C/IOPORT,MODEL,FACTR
C
C   Initialize the pattern normalization factor, "AMPNORM", the maximum
C   number of digits to the right of the decimal place for the linear axis
C   labeling, and the size in inches of the X and Y axes.
C
      AMPNORM=NUMBEAM*NXEL*NYEL*DIPPATM(KO,DIPLNG)
      IF(DGNDPLN .GT. 0.0)AMPNORM=2.0*AMPNORM
      MAXDIG=4
      XSIZE=5.0
      YSIZE=3.0
C
    1 CONTINUE
C
C   Initialize the PLOT88 software.  For plots on the HP Laser printer or the
C   HP7470A plotter the plots are rotated so the Y-axis is along the long edge
C   of the paper.
C
      CALL GRAFHRD
      CALL PLOTS(0,IOPORT,MODEL)
      CALL COMPLX
      IF(MODEL.EQ.96 .OR. MODEL.EQ.99)THEN
         CALL PLOT(1.0,1.0,-3)
         BOXDIR(1)=1.0
         BOXDIR(2)=0.0
         BOXDIR(3)=0.0
      ELSEIF(MODEL.EQ.20 .OR. MODEL.EQ.62)THEN
         CALL PLOT(1.0,XSIZE+1.0,-3)
         BOXDIR(1)=0.0
         BOXDIR(2)=-1.0
         BOXDIR(3)=-90.0
      ENDIF
C
C   Get the constant elevation angle desired for the azimuth cut.
C
      OFFSET=0
      DO 2 I=1,NUMBEAM,1
         CALL NUMCHAR(ELBEAM(I),-2,OFFSET,AXISLBL(1),NCHAR)
         OFFSET=NCHAR+1
```

```
            AXISLBL(1)(OFFSET:)=','
    2 CONTINUE
      WRITE(*,100)AXISLBL(1)(1:NCHAR)
  100 FORMAT(/1X,'Beginning AZIMUTH pattern plot.  Please enter the',
     $  1X,'ELEVATION angle desired'/1X,'for the cut (ELBeam=',A,'): '\)
      READ(*,*,IOSTAT=IOSTAT)ELCON
      DO WHILE (IOSTAT .NE. 0)
         CALL READERR
         WRITE(*,100)AXISLBL(1)(1:NCHAR)
         READ(*,*,IOSTAT=IOSTAT)ELCON
      END DO
      WRITE(*,101)'AZIMUTH'
  101 FORMAT(/5X,'Please wait - Computing the ',A,' pattern...')
C
C   Compute the azimuth pattern.
C
      AZELMX=90.0
      AZELMN=-90.0
      DAZEL=(AZELMX-AZELMN)/(NPTS-1)
      DO 3 IPT=1,NPTS,1
         AZEL(IPT)=(IPT-1)*DAZEL+AZELMN
         CALL AZEL2K(AZEL(IPT),ELCON,PANTILT,KDIR)
         AMP(IPT)=CABS(DIPPAT(KDIR,DIPTILT)*
     $      ARAFACT(K0,KDIR,DGNDPLN,XLOCATE,YLOCATE,DRVCURR,NXEL,
     $      NXELMX,NYEL))
    3 CONTINUE
C
C   Plot the db amplitude of the azimuth pattern.
C
      WRITE(*,102)'AZIMUTH'
  102 FORMAT('+',4X,'Please wait - Generating the ',A,' pattern',
     $ 1X,'plot... ')
      CALL DBAMP(AMP,NPTS,1,NPTS,AMPNORM,-30.0,20.0)
      AXISLBL(1)='Azimuth Angle (degrees)'
      AXISLBL(2)='Amplitude (dB)'
      AMAX=0.0
      AMIN=-30.0
      ADEL=10.0
      DAZEL=45.0
      CALL GRAPH(AZEL,AMP,NPTS,NPTS,1,XSIZE,YSIZE,BOXDIR,AZELMX,AZELMN,
     $  AMAX,AMIN,DAZEL,ADEL,AXISLBL,1,0)
C
      AXISLBL(1)='Elevation Angle of the Cut='
      CALL NUMCHAR(ELCON,-5,27,AXISLBL(1),NCHAR)
      AXISLBL(1)(NCHAR+1:)=' deg.'
      XPAGE=-3.2*BOXDIR(2)
      YPAGE=3.2*BOXDIR(1)
      CALL SYMBOL(XPAGE,YPAGE,0.12,AXISLBL(1),BOXDIR(3),NCHAR+5)
      AXISLBL(1)='AmpNorm='
      WRITE(AXISLBL(1)(9:22),'(1PE14.7)')AMPNORM
      XPAGE=-3.4*BOXDIR(2)
      YPAGE=3.4*BOXDIR(1)
      CALL SYMBOL(XPAGE,YPAGE,0.12,AXISLBL(1),BOXDIR(3),22)
C
C   If displaying plot on terminal print message on screen telling how
C   to proceed, terminate the plot and set up for elevation cut.  If
C   plotting a hardcopy (e.g. HP Laser printer or HP 7470A plotter) reset
C   the plot origin to plot the elevation cut above the azimuth cut.
C
      IF(MODEL.EQ.96 .OR. MODEL.EQ.99)THEN
         AXISLBL(1)='Strike Any Key To Continue'
         IF(MODEL .EQ. 96)CALL COLOR(4,IERR)
         CALL SYMBOL(1.5,4.0,0.12,AXISLBL(1),0.0,26)
         CALL PLOT(0.0,0.0,-999)
         CALL COMPLX
         CALL PLOT(1.0,1.0,-3)
      ELSE
         CALL PLOT(YSIZE+1.5,0.0,-3)
      ENDIF
```

```
C
C    Get the constant azimuth angle desired for the elevation cut.
C
        OFFSET=0
        DO 4 I=1,NUMBEAM,1
           CALL NUMCHAR(AZBEAM(I),-2,OFFSET,AXISLBL(1),NCHAR)
           OFFSET=NCHAR+1
           AXISLBL(1)(OFFSET:)=','
     4  CONTINUE
        WRITE(*,103)'+'//CHAR(7),AXISLBL(1)(1:NCHAR)
   103 FORMAT(A,'Beginning ELEVATION pattern plot.  Please enter',
      $  ' the AZIMUTH angle desired'/1X,'for the cut (AZBeam=',A,'): '\)
        READ(*,*,IOSTAT=IOSTAT)AZCON
        DO WHILE (IOSTAT .NE. 0)
           CALL READERR
           WRITE(*,103)'0',AXISLBL(1)(1:NCHAR)
           READ(*,*,IOSTAT=IOSTAT)AZCON
        END DO
        WRITE(*,101)'ELEVATION'
C
C    Compute the elevation pattern.
C
        DO 5 IPT=1,NPTS,1
           CALL AZEL2K(AZCON,AZEL(IPT),PANTILT,KDIR)
           AMP(IPT)=CABS(DIPPAT(KDIR,DIPTILT)*
      $       ARAFACT(KO,KDIR,DGNDPLN,XLOCATE,YLOCATE,DRVCURR,NXEL,
      $       NXELMX,NYEL))
      5 CONTINUE
C
C    Plot the db amplitude of the elevation pattern.
C
        WRITE(*,102)'ELEVATION'
        CALL DBAMP(AMP,NPTS,1,NPTS,AMPNORM,-30.0,20.0)
        AXISLBL(1)='Elevation Angle (degrees)'
        AMAX=0.0
        AMIN=-30.0
        ADEL=10.0
        DAZEL=45.0
        CALL GRAPH(AZEL,AMP,NPTS,NPTS,1,XSIZE,YSIZE,BOXDIR,AZELMX,AZELMN,
      $  AMAX,AMIN,DAZEL,ADEL,AXISLBL,1,0)
C
        AXISLBL(1)='Azimuth Angle of the Cut='
        CALL NUMCHAR(AZCON,-5,25,AXISLBL(1),NCHAR)
        AXISLBL(1)(NCHAR+1:)=' deg.'
        XPAGE=-3.2*BOXDIR(2)
        YPAGE=3.2*BOXDIR(1)
        CALL SYMBOL(XPAGE,YPAGE,0.12,AXISLBL(1),BOXDIR(3),NCHAR+5)
        AXISLBL(1)='AmpNorm='
        WRITE(AXISLBL(1)(9:22),'(1PE14.7)')AMPNORM
        XPAGE=-3.4*BOXDIR(2)
        YPAGE=3.4*BOXDIR(1)
        CALL SYMBOL(XPAGE,YPAGE,0.12,AXISLBL(1),BOXDIR(3),22)
C
C    If displaying plot on terminal print message on screen telling how
C    to proceed.
C
        IF(MODEL.EQ.96 .OR. MODEL.EQ.99)THEN
           AXISLBL(1)='Strike Any Key To Continue'
           IF(MODEL .EQ. 96)CALL COLOR(4,IERR)
           CALL SYMBOL(1.5,4.0,0.12,AXISLBL(1),0.0,26)
           model=-1
        ENDIF
C
C    Close PLOT88 file.
C
        CALL CLOSPLT
C
C    See if another cut is desired.
C
```

```
      if(YesNoTF('Do you want to see another cut'))go to 1
C
      RETURN
      END
```

```
$PAGESIZE:80
$LINESIZE:132
$MESSAGE:'Compiling AZEL2K'
$SUBTITLE: '    SUBROUTINE AZEL2K'
$PAGE
$NOTRUNCATE
      SUBROUTINE AZEL2K(AZ,EL,ELTILT,K)
C
C   LAST MODIFIED 11/01/88.  BY R. WILSON.
C   ************** 11/02/88.  BY R. WILSON.
C        1) CHANGED K(3) COMPUTATION TO AVOID PROBLEMS WITH NEGATIVE
C           SQUARE ROOT ARGUMENTS RESULTING FROM ROUNDOFF ERRORS.
C   ************** 11/08/88.  BY R. WILSON.
C        1) ADDED ROTATING THE COORDINATE SYSTEM IN ELEVATION BY THE
C           ANGLE "ELTILT" DEGREES BEFORE EXITING.
C   **************  3/14/89.  BY R. WILSON.
C        1) ADDED CODE TO MAKE POSITIVE "AZ" TO BE FROM THE POSITIVE "Z"
C           AXIS TOWARD THE NEGATIVE "X" AXIS.  THIS MAKES THE DEFINITION
C           OF AZIMUTH AGREE WITH THE 'SURVEYORS' DEFINITION.
C
C   "AZEL2K" COMPUTES THE DIRECTION COSINES "KX", "KY", AND "KZ" FOR A VECTOR
C   POINTING IN THE DIRECTION SPECIFIED BY "AZ" AND "EL".  "AZ" AND "EL" ARE
C   ASSUMED TO BE IN DEGREES AND ARE INPUTS TO "AZEL2K".  THE COORDINATE SYSTEM
C   ASSUMED IS WITH THE "Y-Z" PLANE THE VERTICAL PLANE AND THE "X-Z" PLANE THE
C   HORIZONTAL PLANE.  "AZ" IS ASSUMED TO BE SPECIFIED IN THE "X-Z" PLANE
C   MEASURED FROM THE "Z" AXIS AND POSITIVE MOVING TOWARD THE NEGATIVE "X" AXIS
C   (SEE ABOVE).  "EL" IS ASSUMED TO BE MEASURED FROM THE "X-Z" PLANE AND
C   POSITIVE MOVING TOWARD THE POSITIVE "Y" AXIS.  "KX", "KY", AND "KZ" ARE
C   RETURNED IN THE VECTOR "K" WITH "K(1)"="KX", "K(2)"="KY" AND "K(3)"="KZ".
C   BEFORE RETURNING THE COORDINATE SYSTEM IS ROTATED IN ELEVATION (ABOUT THE X
C   AXIS) BY THE ANGLE "ELTILT".
C
      REAL K(3),KY,KZ
C
C   COMPUTE THE DIRECTION COSINES.
C
      K(1)=-COSD(EL)*SIND(AZ)
      KY=SIND(EL)
      KZ=COSD(EL)*COSD(AZ)
C
C   ROTATE THE COORDINATE SYSTEM AROUND THE X AXIS BY "ELTILT" MEASURED FROM
C   THE Z AXIS.
C
      COSTILT=COSD(ELTILT)
      SINTILT=SIND(ELTILT)
      K(2)=KY*COSTILT-KZ*SINTILT
      K(3)=KY*SINTILT+KZ*COSTILT
      RETURN
      END
```

```
$PAGESIZE:80
$LINESIZE:132
$MESSAGE:'Compiling CSOLVE'
$SUBTITLE: '    SUBROUTINE CSOLVE'
$PAGE
$NOTRUNCATE
      SUBROUTINE CSOLVE(A,MXDIMA,N,Y,PARPIV,P,X)
C
C   LAST MODIFIED 12/01/85.  BY R. WILSON.
C   *************  8/30/88.  BY R. WILSON.
C         1) CHANGED SMALLNESS TEST IN SEARCHING FOR NON-ZERO ELEMENT IN
C            A COLUMN (TEST FOR .LE. EPSILON INSTEAD OF .NE. 0.0)
C         2) ADDED CHECK FOR SMALLNESS OF "A(ROW,N)" BEFORE USING AS DIVISOR
C            IN BACK SUBSTITUTION.
C
C   "CSOLVE" COMPUTES THE SOLUTION OF SIMULTANEOUS LINEAR EQUATIONS OF THE FORM
C   Y=AX USING GAUSS-JORDAN ELIMINATION WITH AN OPTION TO PARTIAL PIVOTING.
C
C   DEFINITION OF PARAMETERS.
C
C   "A" IS THE COMPLEX SYSTEM MATRIX.  THE SYSTEM MATRIX IS ASSUMED TO BE A
C   SQUARE MATRIX OF SIZE "N" BY "N".  "CSOLVE" DESTROYS THE "A" MATRIX.  "A" IS
C   AN INPUT TO "CSOLVE".
C
C   "MXDIMA" IS THE ROW DIMENSION OF THE ARRAY IN THE CALLING PROGRAM
C   CORRESPONDING TO THE "A" MATRIX.  IT IS ONLY USED IN "CSOLVE" FOR
C   DIMENSIONING "A".  "MXDIMA" IS AN INPUT TO "CSOLVE".
C
C   "N" IS THE NUMBER OF ROWS AND COLUMNS IN THE "A" MATRIX AND THE NUMBER
C   OF ROWS IN THE "X" AND "Y" COLUMN VECTORS.  IT MUST BE .LE. "MXDIMA".
C   "CSOLVE" CHECKS FOR THIS CONDITION AND TERMINATES THE PROGRAM WITH AN ERROR
C   MESSAGE IF IT IS NOT MET.  "N" IS AN INPUT TO "CSOLVE".
C
C   "Y" IS AN COMPLEX COLUMN ARRAY OF SIZE "N".  IT CORRESPONDS TO THE SYSTEM
C   OUTPUT VECTOR.  "CSOLVE" DESTROYS THE "Y" MATRIX.  IT IS AN INPUT TO
C   "CSOLVE".
C
C   "PARPIV" IS A LOGICAL VARIABLE INDICATING WHETHER PARTIAL PIVIOTING IS
C   TO BE USED.  IF "PARPIV" IS .TRUE. THEN EACH COLUMN IS SEARCHED FOR THE
C   MAXIMUM ELEMENT WHEN SELECTING THE PIVOT ELEMENT.  IF "PARPIV" IS .FALSE.
C   THE FIRST NON-ZERO ELEMENT IN THE COLUMN IS USED AS THE PIVOT.  "PARPIV" IS
C   AN INPUT TO "CSOLVE".
C
C   "P" IS COLUMN VECTOR OF SIZE "N" USED AS A POINTER VECTOR USED FOR
C   EXCHANGING ROWS IN THE ELIMINATION ALGORITHM.  RATHER THAN ACTUALLY MOVING
C   DATA AROUND WHEN ROWS ARE EXCHANGED THE CORRESPONDING ELEMENTS IN "P" ARE
C   EXCHANGED.  "P" IS INITIALIZED BY "CSOLVE" AND THEN CHANGED BY "CSOLVE" IF
C   NECESSARY.  NORMALLY THE CALLING PROGRAM MAKES NO USE OF "P" AFTER CALLING
C   "CSOLVE" AND THE CALLING ROUTINE DOES NOT INITIALIZE "P".  "P" IS INCLUDED
C   IN THE ARGUMENT LIST SO IT CAN BE AN ADJUSTABLE DIMENSIONED ARRAY.  IN THE
C   CALLING PROGRAM "P" SHOULD BE DIMENSIONED "MXDIMA", THAT IS, IT SHOULD HAVE
C   THE SAME DIMENSION AS THE ROW DIMENSION OF THE "A" MATRIX.
C
C   "X" IS A COMPLEX COLUMN VECTOR OF SIZE "N".  IT CORRESPONDS TO THE SYSTEM
C   INPUT VECTOR.  IT IS AN OUTPUT FROM "CSOLVE".
C
      PARAMETER(EPSILON=1.0E-12)
      COMPLEX A(MXDIMA,N),X(N),Y(N),AM
      INTEGER P(N),ROW,PIVROW
      LOGICAL PARPIV
C
C   CHECK FOR CORRECT INPUTS.
C
      IF(N .GT. MXDIMA)THEN
         CALL BEEP
         WRITE(*,100)N,MXDIMA
  100    FORMAT(/1X,'***** ERROR - Program terminated in CSOLVE.'/
     $      7X,'N(=',I5,') .GT. MXDIMA(=',I5,').')
         STOP
```

```
      ENDIF
C
C   INITIALIZE PIVOT POINTER VECTOR, P.
C
      DO 1 I=1,N,1
         P(I)=I
    1 CONTINUE
C
C   UPPER TRIANGULARIZE A MATRIX.
C
      NM1=N-1
      DO 7 KCOL=1,NM1,1
         KCOLP1=KCOL+1
C
C     SEARCH "KCOL" COLUMN OF "A" FROM THE "P(KCOL)"TH ROW TO THE "P(N)TH"
C     ROW FOR THE FIRST NON-ZERO ELEMENT.
C
         DO 2 J=KCOL,N,1
            NZROW=P(J)
            NZJ=J
            IF(ABS(A(NZROW,KCOL)) .GE. EPSILON)GO TO 3
    2    CONTINUE
         CALL BEEP
         WRITE(*,101)
  101    FORMAT(1X,'***** ERROR - program terminated in CSOLVE.'/
     $      7X,'The "A" matrix is not invertible.')
         STOP
    3    CONTINUE
         IF(PARPIV)THEN
C
C          SEARCH "KCOL" COLUMN OF "A" FROM THE "NZROW"-TH ROW TO THE
C          "P(N)"-TH ROW FOR THE MAXIMUM ELEMENT.
C
            MAXROW=NZROW
            MAXPJ=NZJ
            DO 4 J=NZJ+1,N,1
               ROW=P(J)
               IF(ABS(A(ROW,KCOL)) .GT. ABS(A(MAXROW,KCOL)))THEN
                  MAXROW=ROW
                  MAXPJ=J
               ENDIF
    4       CONTINUE
            J=MAXPJ
            ROW=MAXROW
         ELSE
            J=NZJ
            ROW=NZROW
         ENDIF
C
C     NON-ZERO (AND/OR MAXIMUM) ELEMENT FOUND, USE THIS ROW AS PIVOT.
C     EXCHANGE "P(KCOL)" WITH "P(J)" IN PIVOT POINTER VECTOR TO FACILITATE
C     ARRAY INDEXING LATER
C
         P(J)=P(KCOL)
         P(KCOL)=ROW
C
C     CALCULATE PIVOT MULTIPLIER, "AM", AND SET ARRAY ELEMENTS IN COLUMN
C     "KCOL" TO AM IF THEY HAVEN'T BEEN USED AS A PIVOT ACCORDING TO PIVOT
C     POINTER VECTOR "P".
C
         DO 6 I=KCOLP1,N,1
            PIVROW=P(KCOL)
            ROW=P(I)
            AM=A(ROW,KCOL)/A(PIVROW,KCOL)
            A(ROW,KCOL)=AM
C
C          CALCULATE NEW VALUES OF COEFFCIENTS IN COLUMS "KCOL+1" TO "N" OF
C          THE "A" MATRIX AND ROW "P(I)" OF THE "Y" ARRAY.
C
```

```
            DO 5 J=KCOLP1,N,1
               A(ROW,J)=A(ROW,J)-AM*A(PIVROW,J)
    5       CONTINUE
            Y(ROW)=Y(ROW)-AM*Y(PIVROW)
    6    CONTINUE
    7 CONTINUE
C
C   SOLVE "TRIANGULARIZED" SYSTEM BY BACK SUBSTITION.
C
      ROW=P(N)
      IF(ABS(A(ROW,N)) .LT. EPSILON)THEN
         CALL BEEP
         WRITE(*,101)
         STOP
      ENDIF
      X(N)=Y(ROW)/A(ROW,N)
      DO 9 I=1,NM1,1
         NMI=N-I
         ROW=P(NMI)
         X(NMI)=Y(ROW)
         DO 8 J=1,I,1
            NMIPJ=NMI+J
            X(NMI)=X(NMI)-X(NMIPJ)*A(ROW,NMIPJ)
    8    CONTINUE
         X(NMI)=X(NMI)/A(ROW,NMI)
    9 CONTINUE
      RETURN
      END
```

```
$PAGESIZE:80
$LINESIZE:132
$MESSAGE:'Compiling DIPPATM'
$SUBTITLE: '    FUNCTION DIPPATM'
$PAGE
$NOTRUNCATE
      FUNCTION DIPPATM(K0,DIPLNG)
C
C   LAST MODIFIED 12/11/88.  BY R. WILSON.
C
C   "DIPPAT" IS A COMPLEX FUNCTION WHICH COMPUTES THE FAR-FIELD AMPLITUDE IN
C   THE DIRECTION "K" OF A DIPOLE OF LENGTH "DIPLNG" WHOSE AXIS IS ALONG THE
C   VECTOR "DIPTILT". "DIPPATM" IS THE FIRST ENTRY POINT AND INITIALIZES SOME
C   VALUES NEEDED BY THE ENTRY POINT "DIPPAT".  "DIPPATM" RETURNS THE VALUE
C   OF THE MAGNITUDE OF THE PATTERN AT BROADSIDE.  THE EQUATION IMPLEMENTED IS
C   EQ. 4-62A FROM BALANIS, "ANTENNA THEORY - ANALYSIS AND DESIGN", PP. 120.
C
      PARAMETER(PI=3.1415926535898,PIOVER2=PI/2.0,TWOPI=2.0*PI)
      PARAMETER(ZERO=1.0E-12)
      COMPLEX DIPPAT,DIPAMP
      REAL KDIR(3),DIPTILT(3),K0,DIPLNG,FFRAD,KLOVER2
C
C   INTIALIZE VARIABLES USED BY "DIPPAT".  COMPUTE THE PATTERN MAGNITUDE AT
C   "THETA"=90.0 DEGREES.
C
      KLOVER2=K0*DIPLNG/2.0
      FFRAD=1.0E6
      ETA=377.0
      PHASE=-MOD(K0*FFRAD,TWOPI)+PIOVER2
      COSKL2=COS(KLOVER2)
      DIPAMP=ETA* CEXP(CMPLX(0.0,PHASE))/(TWOPI*FFRAD)
      DIPPATM=ETA*(1.0-COSKL2)/(TWOPI*FFRAD)
      RETURN
C
C   COMPUTE THE PATTERN IN THE DIRECTION "KBEAM".  NOTE: IT IS ASSUMED THAT
C   THE Z-COMPONENT OF THE DIPOLE TILT, "DIPTILT(3)", IS ZERO, SEE "GDTILT".
C
      ENTRY DIPPAT(KDIR,DIPTILT)
      COSTH=KDIR(1)*DIPTILT(1)+KDIR(2)*DIPTILT(2)
      SINTH=SQRT(1.0-COSTH**2)
      IF(ABS(SINTH) .GT. ZERO)THEN
         DIPPAT=DIPAMP*(COS(KLOVER2*COSTH)-COSKL2)/SINTH
      ELSE
         DIPPAT=CMPLX(0.0,0.0)
      ENDIF
      RETURN
      END
```

```
$PAGESIZE:80
$LINESIZE:132
$MESSAGE:'Compiling DRVPNT'
$SUBTITLE: '    SUBROUTINE DRVPNT'
$PAGE
$NOTRUNCATE
      SUBROUTINE DRVPNT(LAMBDA,NXEL,NXELMX,NYEL,XLOCATE,YLOCATE,DIPWID,
     $  DIPLNG,DIPTILT,DGNDPLN,DRVCURR,ZDRVIMP,ZSYSROW,NXYELMX)
C
C    LAST MODIFIED 4/23/89.  BY R. WILSON.
C    ************* 5/08/89.  BY R. WILSON.
C         1) REMOVED COMPUTATION OF DRIVING POINT VOLTAGES.
C    ************* 6/06/89.  BY R. WILSON.
C         1) CORRECTED METHOD OF LOADING IMPEDANCES TO ACCOUNT FOR THE FACT
C            THAT WHEN THE DIPOLES ARE TILTED (NOT 0 OR 90 DEGREES) THE
C            IMPEDANCES ARE FUNCTIONS OF SIGN(IVOL-ICUR)*SIGN(JVOL-JCUR) NOT
C            ABS(IVOL-ICUR)*ABS(JVOL-JCUR).
C
C    "DRVPNT" LOADS THE DRIVING POINT IMPEDANCE ARRAY, "ZDRVIMP", FOR AN ARRAY
C    OF IDENTICAL EQUALLY SPACED DIPOLES.  MAXIMUM USE IS MADE OF SYMMETRY,
C    RESULTING IN IT BEING NECESSARY TO STORE ONLY TWO ROWS OF THE SYSTEM
C    MATRIX (ONLY NECESSARY TO COMPUTE PART OF THE SECOND ROW).
C
C    "LAMBDA" IS THE WAVELENGTH OF THE SOURCE.  THE UNITS OF "LAMBDA" SHOULD
C    BE CONSISTENT WITH THAT OF ALL OTHER LINEAR DIMENSIONS.  IT IS AN INPUT
C    TO "DRVPNT".
C
C    "NXEL" AND "NYEL" ARE THE NUMBER OF DIPOLES ELEMENTS IN THE HORIZONTAL AND
C    VERTICAL DIRECTIONS RESPECTIVELY.  THEY ARE INPUTS TO "DRVPNT".
C
C    "NXELMX" IS THE MAXIMUM NUMBER OF DIPOLES ELEMENTS IN THE HORIZONTAL
C    DIRECTION.  IT IS USED MAINLY FOR DIMENSIONING THE ARRAYS "DRVCURR",
C    AND "ZDRVIMP".  IT IS AN INPUT TO "DRVPNT".
C
C    "XLOCATE" AND "YLOCATE" ARE ONE DIMENSIONAL ARRAYS CONTAINING THE
C    HORIZONTAL AND VERTICAL LOCATIONS OF EACH DIPOLE ELEMENT.  FOR EXAMPLE, THE
C    LOCATION OF THE (2,4)TH ELEMENT IN THE ARRAY IS (XLOCATE(2),YLOCATE(4)).
C    "XLOCATE" AND "YLOCATE" ARE ASSUMED TO HAVE UNITS CONSISTENT WITH ALL OTHER
C    LINEAR DIMENSIONS.  THEY ARE INPUTS TO "DRVPNT".
C
C    "DIPWID" IS THE WIDTH OF THE CONDUCTORS MAKING UP EACH DIPOLE IN THE ARRAY.
C    THE UNITS FOR "DIPWID" SHOULD BE CONSISTENT WITH ALL OTHER LINEAR
C    DIMENSIONS.   "DIPWID" IS AN INPUT TO "DRVPNT".
C
C    "DIPLNG" IS THE LENGTH OF EACH DIPOLE IN THE ARRAY.  THE UNITS FOR "DIPLNG"
C    SHOULD BE CONSISTENT WITH ALL OTHER LINEAR DIMENSIONS.  "DIPLNG" IS AN
C    INPUT TO "DRVPNT".
C
C    "DIPTILT" IS AN ARRAY CONTAING A UNIT DIRECTION VECTOR SPECIFYING THE
C    DIRECTION OF THE DIPOLES IN THE APERTURE PLANE.  IT SHOULD BE DIMENSIONED
C    THREE (3) IN THE CALLING PROGRAM. SINCE THE Z COMPONENT OF "DIPTILT" WILL
C    ALWAYS BE 0 THE ANGLE OF THE DIPOLES, IN DEGREES WITH RESPECT TO THE
C    APERTURE VERTICAL IS STORED IN "DIPTILT(3)".  "DIPTILT" IS AN INPUT TO
C    "DRVPNT".
C
C    "DGNDPLN", IF .GT.  0.0 SPECIFIES THE DISTANCE FROM THE ARRAY PANEL TO THE
C    GROUND PLANE.  IF "DGNDPLN" IS .LE.  0.0 THEN NO GROUND PLANE IS PRESENT.
C    THE UNITS FOR "DGNDPLN" SHOULD BE CONSISTENT WITH ALL OTHER LINEAR
C    DIMENSIONS.  " DGNDPLN" IS AN INPUT TO "DRVPNT".
C
C    "DRVCURR" IS A TWO DIMENSIONAL COMPLEX ARRAY CONTAINING THE DRIVING POINT
C    CURRENTS FOR EACH OF THE DIPOLES IN THE ARRAY.  IT IS AN INPUT TO "DRVPNT".
C
C    "ZSYSROW" IS A COMPLEX ARRAY WHICH HOLDS THE FIRST AND "NXEL"TH ROW OF THE
C    SYSTEM IMPEDANCE MATRIX.  IT IS IN THE ARGUMENT LIST SO THAT ITS SIZE CAN
C    BE MADE ADJUSTABLE.  IT SHOULD BE DIMENSIONED ("NXYELMX",2).  IT IS AN
C    OUTPUT FROM "DRVPNT".
C
C    "NXYELMX" IS THE ROW DIMENSION OF "ZSYSROW".  IT SHOULD .GE. "NXEL"*"NYEL".
```

```
C   IT IS AN INPUT TO "DRVPNT".
C
      COMPLEX ZDRVIMP(NXELMX,NYEL),ZSYSROW(NXYELMX,2)
      COMPLEX DRVCURR(NXELMX,NYEL),ZGRND,DRVVLT
      REAL XLOCATE(NXEL),YLOCATE(NYEL),DIPTILT(3),DIPRAD(2)
      REAL LAMBDA,LAMBDAI,DIPLNG,DIPLNGW,DIPWID,DGNDPLN
      LOGICAL UGNDPLN
C
C   CHECK FOR CORRECT INPUTS.
C
      NXYEL=NXEL*NYEL
      IF(NXYEL .GT. NXYELMX)THEN
         CALL BEEP
         WRITE(*,101)NXYEL,NXYELMX
  101    FORMAT(/1X,'***** ERROR - program terminated in DRVIMP.'/
     $     7X,'NXYEL(=',I3,') .GT. NXYELMX(=',I3,').')
         STOP
      ENDIF
C
C   COMPUTE THE DIPOLE RADII, "DIPRAD(I)", FROM THE DIPOLE WIDTHS, "DIPWID"
C   (THE CONVERSION FROM THE WIDTH OF A FLAT CONDUCTOR TO AN EQUIVALENT RADIUS
C   OF A ROUND CONDUCTOR IS FOUND IN C.  A.  BALANIS, "ANTENNA THEORY, ANALYSIS
C   AND DESIGN", PP.  337-339).  NORMALIZE ALL LINEAR DIMENSIONS TO WAVELENGTH,
C   "LAMBDA".  INITIALIZE "UGNDPLN" AND "ZIMAGE" FOR GROUND PLANE COMPUTATION.
C
      LAMBDAI=1.0/LAMBDA
      DIPRAD(1)=DIPWID*LAMBDAI*0.25
      DIPRAD(2)=DIPRAD(1)
      DIPLNGW=DIPLNG*LAMBDAI
      DO 1 I=1,NXEL,1
         XLOCATE(I)=XLOCATE(I)*LAMBDAI
    1 CONTINUE
      DO 2 I=1,NYEL,1
         YLOCATE(I)=YLOCATE(I)*LAMBDAI
    2 CONTINUE
      UGNDPLN=DGNDPLN .GT. 0.0
      ZIMAGE=-2.0*DGNDPLN*LAMBDAI
C
C   LOAD THE FIRST ROW/COLUMN OF THE SYSTEM IMPEDANCE MATRIX, "ZSYSROW".
C   IF A GROUND PLANE IS PRESENT ADJUST FOR ITS EFFECT.
C
      CALL MOM(1,0.0,0.0,0.0,DIPLNGW,DIPLNGW,DIPRAD,ZSYSROW(1,1))
      IF(UGNDPLN)THEN
         CALL MOM(2,0.0,ZIMAGE,0.0,DIPLNGW,DIPLNGW,DIPRAD,ZGRND)
         ZSYSROW(1,1)=ZSYSROW(1,1)-ZGRND
      ENDIF
      IF(NXYEL .GT. 1)THEN
         IF(NXEL.GT.1  .AND. NYEL.GT.1)THEN
            NUNIQRW=2
         ELSE
            NUNIQRW=1
         ENDIF
         ICURX=1
         DO 4 IUNIQRW=1,NUNIQRW,1
            XCUR=XLOCATE(ICURX)
            YCUR=YLOCATE(1)
            DO 3 I=2,NXYEL,1
               JVOLY=(I-1)/NXEL+1
               IVOLX=I-(JVOLY-1)*NXEL
               IF(IUNIQRW.EQ.1 .OR.
     $            (IVOLX-ICURX)*(JVOLY-1) .LT. 0)THEN
                  XDIFF=XLOCATE(IVOLX)-XCUR
                  YDIFF=YLOCATE(JVOLY)-YCUR
C
C                ROTATE COORDINATES TO ALLOW "MOM" TO SEGMENT ALONG THE X-AXIS.
C
                  XOFFSET=XDIFF*DIPTILT(1)+YDIFF*DIPTILT(2)
                  YOFFSET=-XDIFF*DIPTILT(2)+YDIFF*DIPTILT(1)
C
```

```
C              COMPUTE THE MUTUAL IMPEDANCE BETWEEN REAL DIPOLES.
C
                 CALL MOM(2,YOFFSET,0.0,XOFFSET,DIPLNGW,DIPLNGW,DIPRAD,
     $             ZSYSROW(I,IUNIQRW))
C
C              ADJUST THE IMPEDANCE FOR PRESENCE OF A GROUND PLANE.
C
                 IF(UGNDPLN)THEN
                    CALL MOM(2,YOFFSET,ZIMAGE,XOFFSET,DIPLNGW,DIPLNGW,
     $                DIPRAD,ZGRND)
                    ZSYSROW(I,IUNIQRW)=ZSYSROW(I,IUNIQRW)-ZGRND
                 ENDIF
              ENDIF
    3      CONTINUE
           ICURX=NXEL
    4   CONTINUE
      ENDIF
C
C      COMPUTE THE DRIVING POINT VOLTAGES AND DRIVING POINT IMPEDANCES.
C      ASSUME ALL DIPOLES ARE IDENTICAL AND SPACING BETWEEN DIPOLES IS
C      IDENTICAL.
C
      DO 6 I=1,NXYEL,1
         JVOLY=(I-1)/NXEL+1
         IVOLX=I-(JVOLY-1)*NXEL
         DRVVLT=CMPLX(0.0,0.0)
         DO 5 J=1,NXYEL,1
            JCURY=(J-1)/NXEL+1
            ICURX=J-(JCURY-1)*NXEL
            IDIFX=IVOLX-ICURX
            JDIFY=JVOLY-JCURY
            IF((IDIFX*JDIFY) .GE. 0)THEN
               IFSTROW=ABS(IDIFX)+ABS(JDIFY)*NXEL+1
               DRVVLT=DRVVLT+ZSYSROW(IFSTROW,1)*DRVCURR(ICURX,JCURY)
            ELSE
               IFSTROW=-ABS(IDIFX)+ABS(JDIFY)*NXEL+NXEL
               DRVVLT=DRVVLT+ZSYSROW(IFSTROW,2)*DRVCURR(ICURX,JCURY)
            ENDIF
    5    CONTINUE
         ZDRVIMP(IVOLX,JVOLY)=DRVVLT/DRVCURR(IVOLX,JVOLY)
    6 CONTINUE
C
C     DENORMALIZE THE LOCATION ARRAYS.
C
      DO 7 I=1,NXEL,1
         XLOCATE(I)=XLOCATE(I)*LAMBDA
    7 CONTINUE
      DO 8 I=1,NYEL,1
         YLOCATE(I)=YLOCATE(I)*LAMBDA
    8 CONTINUE
      RETURN
      END
```

```
$PAGESIZE:80
$LINESIZE:132
$MESSAGE:'Compiling DRWARA'
$SUBTITLE: '    SUBROUTINE DRWARA'
$PAGE
$NOTRUNCATE
      SUBROUTINE DRWARA(UNITS,XSIZE,YSIZE,NXEL,NYEL,DIPLNG,DIPTILT,
     $  LFILNET,SFILNET,DGNDPLN,XLOCATE,YLOCATE,ZDRVIMP,NXELMX)
C
C   LAST MODIFIED  9/27/88.  BY R. WILSON.
C   ************** 11/08/88.  BY R. WILSON.
C        1) Added handling of panel tilt.
C   ************** 11/15/88.  BY R. WILSON.
C        1) Plotting of title block now handled by "ARATITL".
C   ************** 11/22/88.  BY R. WILSON.
C        1) Changed so that if array panel is rectangular the
C           box representing the panel edge will be rectangular.
C   ************** 12/11/88.  BY R. WILSON.
C        1) Added handling of dipole tilt.
C   **************  4/04/89.  BY R. WILSON.
C        1) Added handling of a ground plane.
C   **************  8/08/89.  BY R. WILSON.
C        1) Changed the plotting of the impedance values so that they are
C           staggered, thus preventing overwriting.
C
C   "DRWARA" draws a scaled version of the dipole array, showing the location
C   of each dipole.
C
      COMPLEX ZDRVIMP(NXELMX,NYEL)
      REAL XLOCATE(NXEL),YLOCATE(NYEL),XORIGIN(2),DIPTILT(3)
      REAL BOXDIR(3),MAXDIM,LBLLOC
      INTEGER OFFSET,MAXDIG
      CHARACTER AXISLBL(2)*80,UNITS*2,LFILNET*1
      COMMON/LINAXC/MAXDIG
      COMMON/PLOT88C/IOPORT,MODEL
C
C   Initialize the maximum number of digits to the right of the decimal
C   place shown on the linear axis labels, "MAXDIG", the size, in inches, of
C   the X and Y axes, "XSIZEPL" and "YSIZEPL", and the direction vector for
C   the X axis of the box, "BOXDIR".
C
      MAXDIG=2
      MAXDIM=MAX(XSIZE,YSIZE)
      XSIZEPL=5.0*XSIZE/MAXDIM
      YSIZEPL=5.0*YSIZE/MAXDIM
      BOXDIR(1)=1.0
      BOXDIR(2)=0.0
      BOXDIR(3)=0.0
C
C   Initialize the PLOT88 software.
C
      CALL GRAFHRD
      CALL PLOTS(0,IOPORT,MODEL)
      IF(IOPORT.EQ.96 .OR. IOPORT.EQ.99)CALL FACTOR(0.8)
      CALL COMPLX
      CALL PLOT(1.5,1.5,-3)
      WRITE(*,100)
  100 FORMAT(/5X,'Please wait - generating the array schematic...')
C
C   Draw the outside border of the array.
C
      AXISLBL(1)='X ('//UNITS//'.)'
      AXISLBL(2)='Y ('//UNITS//'.)'
      DELTAX=XSIZE/2
      DELTAY=YSIZE/2
      CALL AXISXY(1,XSIZEPL,YSIZEPL,BOXDIR,XSIZE,0.0,YSIZE,0.0,DELTAX,
     $  DELTAY,AXISLBL,XSCALE,YSCALE)
C
C   If a ground plane is used plot legend showing its location.
```

```
C
      IF(DGNDPLN .GT. 0.0)THEN
         AXISLBL(1)='A ground plane is to be used'
         OFFSET=29
         CALL NUMCHAR(DGNDPLN,-3,OFFSET,AXISLBL(1),NCHAR)
         AXISLBL(1)(NCHAR+1:)=UNITS//'. from the array panel'
         NCHAR=NCHAR+24
         XPAGE=-(YSIZEPL+0.2)*BOXDIR(2)
         YPAGE=(YSIZEPL+0.2)*BOXDIR(1)
         CALL SYMBOL(XPAGE,YPAGE,0.12,AXISLBL(1),BOXDIR(3),NCHAR)
      ENDIF
C
C  Draw the dipoles.
C
      DIPLNG2=(DIPLNG/2)*XSCALE
      DIPLNGS=DIPLNG*XSCALE
      HGHTX=MIN(0.12,DIPLNG2)
      ANGLE=90.0-DIPTILT(3)
      DO 1 J=1,NYEL,1
         XORIGIN(2)=-DIPLNG2*DIPTILT(2)+YLOCATE(J)*YSCALE
         IF( (J/2)*2 .EQ. J)THEN
            LBLLOC=-0.2
            DELTA=-0.2
         ELSE
            LBLLOC=-0.4
            DELTA=0.2
         ENDIF
         DO 2 I=1,NXEL,1
            LBLLOC=LBLLOC+DELTA
            DELTA=-DELTA
            XORIGIN(1)=-DIPLNG2*DIPTILT(1)+XLOCATE(I)*XSCALE
C
C          Left end of dipole (assuming horizontal dipoles).
C
            CALL SYMBOL(XORIGIN(1),XORIGIN(2),HGHTX,CHAR(13),ANGLE,-1)
            CALL PLOT(XORIGIN(1),XORIGIN(2),3)
C
C          Right end of dipole (assuming horizontal dipoles).
C
            XPAGE=DIPLNGS*DIPTILT(1)+XORIGIN(1)
            YPAGE=DIPLNGS*DIPTILT(2)+XORIGIN(2)
            CALL PLOT(XPAGE,YPAGE,2)
            CALL SYMBOL(XPAGE,YPAGE,HGHTX,CHAR(13),ANGLE,-1)
C
C          Indicate the center of the dipole.
C
            XPAGE=DIPLNG2*DIPTILT(1)+XORIGIN(1)
            YPAGE=DIPLNG2*DIPTILT(2)+XORIGIN(2)
            CALL SYMBOL(XPAGE,YPAGE,HGHTX,CHAR(4),ANGLE,-1)
C
C         Label with element driving point impedance.
C
            AXISLBL(1)='Z='
            CALL NUMCHAR(REAL(ZDRVIMP(I,J)),-1,2,AXISLBL(1),NCHAR)
            AXISLBL(1)(NCHAR+1:)='+j'
            OFFSET=NCHAR+2
            CALL NUMCHAR(AIMAG(ZDRVIMP(I,J)),-1,OFFSET,
     $        AXISLBL(1),NCHAR)
            IF(AXISLBL(1)(OFFSET+1:OFFSET+1) .EQ. '-')THEN
               AXISLBL(1)(OFFSET-1:OFFSET+1)='-j'//CHAR(0)
            ENDIF
            CALL TITLE(AXISLBL(1),ANGLE,1,LBLLOC,0.1,DIPLNGS,XORIGIN)
    2    CONTINUE
    1 CONTINUE
C
C   Show location of the filter network.
C
      AXISLBL(1)='Filter Network'
      IF(LFILNET .EQ. 'B')THEN
```

```
            FSPACE=SFILNET*YSCALE
            CALL PLOT(0.0,FSPACE,3)
            CALL PLOT(XSIZEPL,FSPACE,2)
            XORIGIN(1)=0.0
            XORIGIN(2)=FSPACE/2.0
            CALL TITLE(AXISLBL(1),0.0,1,0.0,0.12,XSIZEPL,XORIGIN)
         ELSEIF(LFILNET .EQ. 'T')THEN
            FSPACE=SFILNET*YSCALE
            XORIGIN(2)=YSIZEPL-FSPACE
            CALL PLOT(0.0,XORIGIN(2),3)
            CALL PLOT(XSIZEPL,XORIGIN(2),2)
            XORIGIN(1)=0.0
            XORIGIN(2)=YSIZEPL-FSPACE/2.0
            CALL TITLE(AXISLBL(1),0.0,1,0.0,0.12,XSIZEPL,XORIGIN)
         ELSEIF(LFILNET .EQ. 'L')THEN
            FSPACE=SFILNET*XSCALE
            CALL PLOT(FSPACE,0.0,3)
            CALL PLOT(FSPACE,YSIZEPL,2)
            XORIGIN(1)=FSPACE/2.0
            XORIGIN(2)=0.0
            CALL TITLE(AXISLBL(1),90.0,1,0.0,0.12,YSIZEPL,XORIGIN)
         ELSEIF(LFILNET .EQ. 'R')THEN
            FSPACE=SFILNET*XSCALE
            XORIGIN(1)=XSIZEPL-FSPACE
            CALL PLOT(XORIGIN(1),0.0,3)
            CALL PLOT(XORIGIN(1),YSIZEPL,2)
            XORIGIN(1)=XSIZEPL-FSPACE/2.0
            XORIGIN(2)=0.0
            CALL TITLE(AXISLBL(1),90.0,-1,0.0,0.12,YSIZEPL,XORIGIN)
         ENDIF
C
C   If displaying plot on terminal print message on screen telling how
C   to proceed.
C
         IF(MODEL.EQ.96 .OR. MODEL.EQ.99)THEN
            AXISLBL(1)='Strike Any Key to Continue'
            IF(MODEL .EQ. 96)CALL COLOR(4,IERR)
            call factor(1.0)
c            CALL SYMBOL(1.5,-1.25,0.12,AXISLBL(1),0.0,26)
            call symbol(1.2,-1.0,0.12,AXISLBL(1),0.0,26)
            model=-1
         ENDIF
C
C   Close PLOT88 files.
C
         CALL CLOSPLT
C
         RETURN
         END
```

```
$PAGESIZE:80
$LINESIZE:132
$MESSAGE:'Compiling GDLNGTH'
$SUBTITLE: '    SUBROUTINE GDLNGTH'
$PAGE
$NOTRUNCATE
      SUBROUTINE GDLNGTH(LAMBDA,NXEL,NYEL,DX,DY,DIPWID,DIPTILT,
     $  XACTIVE,YACTIVE,DIPLNG)
C
C   LAST MODIFIED 12/14/88.  BY R. WILSON.
C   *************  4/03/89.  BY R. WILSON.
C        1) CORRECTED ERROR WHICH WAS CAUSED BY "LNGTHX" AND/OR  "LNGTHY"
C           BEING NEGATIVE.  THIS HAPPENED IF "DIPTILT(3)" ENDED IN A QUADRANT
C           OTHER THAN THE FIRST.
C
C   "GDLNGTH" DETERMINES THE LENGTHS OF THE DIPOLES IN THE ARRAY.   "GDLNGTH"
C   STANDS FOR Get_Dipole_LeNGTH.
C
C   "LAMBDA" IS THE WAVELENGTH OF THE SOURCE.  IT IS AN INPUT TO "GDLNGTH".
C
C   "NXEL", AND "NYEL" ARE THE NUMBER OF ELEMENTS HORIZONTALLY AND VERTICALLY
C   IN THE ARRAY.  THEY ARE INPUTS TO "GDLNGTH".
C
C   "DX" AND "DY" ARE THE HORIZONTAL AND VERTICAL SPACING BETWEEN THE CENTERS
C   OF THE DIPOLES IN THE ARRAY.  THEY HAVE THE SAME UNITS AS "LAMBDA" AND ARE
C   INPUTS TO "GDLNGTH".
C
C   "DIPWID" IS THE WIDTH OF THE DIPOLES IN THE ARRAY.  IT HAS THE SAME UNITS AS
C   "LAMBDA" AND IS AN INPUT TO "GDLNGTH".
C
C   "DIPTILT" IS AN ARRAY CONTAINING A UNIT DIRECTION VECTOR SPECIFYING THE
C   DIRECTION OF THE DIPOLES IN THE APERTURE PLANE.  SEE "GDTILT" FOR A
C   DETAILED DESCRIPION.  "DIPTILT" IS AN INPUT TO "GDLNGTH".
C
C   "XACTIVE" AND "YACTIVE" ARE THE HORIZONTAL AND VERTICAL LENGTHS OF THE AREA
C   OF THE APERTURE WHICH CONTAINS THE DIPOLES.  SEE "GSIZE" FOR A DETAILED
C   DESCRIPTION.  THEY ARE INPUTS TO "GDLNGTH".
C
C   "DIPLNG" IS THE LENGTH OF THE DIPOLES IN THE ARRAY.  IT HAS THE SAME UNITS
C   AS "LAMBDA", ETC.  IT IS THE OUTPUT FROM "GDLNGTH".
C
      REAL DIPTILT(3)
      REAL LAMBDA,LNGTHX,LNGTHY
C
C   "RESLNG" IS THE APPROXIMATE RESONANT LENGTH FOR DIPOLES WITH WIDTHS AS
C   RETURNED BY "DIPWID".
C
      RESLNG=0.465*LAMBDA
C
C   COMPUTE THE DIPOLE LENGTH.
C
      IF(NXEL .GT. 1)THEN
         LNGTHX=DX*DIPTILT(1)
      ELSE
         LNGTHX=XACTIVE*DIPTILT(1)
      ENDIF
      IF(NYEL .GT. 1)THEN
         LNGTHY=DY*DIPTILT(2)
      ELSE
         LNGTHY=YACTIVE*DIPTILT(2)
      ENDIF
      DIPLNG=MIN(ABS(LNGTHX)+ABS(LNGTHY)-DIPWID,RESLNG)
      RETURN
      END
```

```
$PAGESIZE:80
$LINESIZE:132
$MESSAGE:'Compiling GDTILT'
$SUBTITLE: '    SUBROUTINE GDTILT'
$PAGE
$NOTRUNCATE
      SUBROUTINE GDTILT(DIPTILT)
C
C   LAST MODIFIED 12/11/88.  BY R. WILSON.
C   *************  4/03/89.  BY R. WILSON.
C        1) Changed prompt for tilt angle to define the meaning of a positive
C           tilt angle.
C   *************  7/20/89.  BY R. WILSON.
C        1) Added trapping of operator data entry errors.
C        2) Previously a typing mistake would terminate the program with a
C           run-time error message and a return to DOS.
C   *************  9/15/89.  BY R. WILSON.
C        1) Added "polarity" in parenthesis to the user prompt as suggested by
C           Joe Zimmerman.
C
C   "GDTILT" prompts for the tilt angle of the dipoles in the array with
C   respect to the panel vertical.  The vector of direction cosines is then
C   computed and stored in "DIPTILT(1)" and "DIPTILT(2)".  The tilt angle
C   itself is stored in "DIPTILT(3)".  "GDTILT"=GetDipoleTILT"
C
      REAL DIPTILT(3)
C
C   Obtain the desired tilt angle.
C
      WRITE(*,100)
  100 FORMAT(/1X,'The dipoles making up the array can be tilted in the',
     $  1X,'aperture.  At the prompt'/1X,'enter the desired dipole',
     $  1X,'tilt angle.  The angle should be specified in degrees'/
     $  1X,'measured from the vertical.  The dipoles will be vertical',
     $  1X,'for a 0 degree tilt'/1X,'angle and horizontal for a 90',
     $  1X,'degree tilt angle.  A positive tilt angle is'/1X,'assumed',
     $  1X,'to be in a COUNTER-CLOCKWISE direction when looking in the',
     $  1X,'same'/1X,'direction as the array.'//5X,'Enter the desired',
     $  1X,'dipole tilt (polarity): '\)
      READ(*,*,IOSTAT=IOSTAT)DIPTILT(3)
      DO WHILE (IOSTAT .NE. 0)
         CALL READERR
         WRITE(*,100)
         READ(*,*,IOSTAT=IOSTAT)DIPTILT(3)
      END DO
C
C   Compute the dipole direction vector.
C
      DIPTILT(1)=SIND(DIPTILT(3))
      DIPTILT(2)=COSD(DIPTILT(3))
      RETURN
      END
```

```
$PAGESIZE:80
$LINESIZE:132
$MESSAGE:'Compiling GDWIDTH'
$SUBTITLE: '    SUBROUTINE GDWIDTH'
$PAGE
$NOTRUNCATE
      SUBROUTINE GDWIDTH(UNITS,LAMBDA,DIPWID)
C
C    LAST MODIFIED 11/17/88.  BY R. WILSON.
C    **************  5/08/89.  BY R. WILSON.
C         1) SPLIT OFF CONVERSION TO INCHES INTO A SEPERATE FUNCTION.
C    **************  5/11/89.  BY R. WILSON.
C         1) CHANGED THE VALUES IN "STDWID" TO MORE STANDARD VALUES.
C
C    "GDWIDTH" RETURNS THE DIPOLE WIDTHS, "DIPWID", IN UNITS OF "UNITS".  IT IS
C    CHOOSEN FROM A SET OF "STANDARD" WIDTHS CONTAINED IN THE ARRAY "STWID".
C    "LAMBDA" IS THE WAVELENGTH OF THE SOURCE IN UNITS OF "UNITS.  "GDWIDTH"
C    STANDS FOR Get_Dipole_WIDTH.
C
      PARAMETER(NUMWID=6)
      REAL STDWID(NUMWID),LAMBDA,DIPWID
      CHARACTER UNITS*2
      DATA STDWID/0.125,0.25,0.375,0.5,0.75,1.00/
C
C    DETERMINE THE CONDUCTOR WIDTH, "DIPWID", TO BE USED FOR THE DIPOLES.
C    THE STANDARD WIDTHS, "STDWID", ARE IN INCHES SO COMPUTE THE INITIAL
C    GUESS IN INCHES.
C
      DIPWID=UNIT2IN(LAMBDA,UNITS)/120.0
C
C    FIND THE CLOSEST STANDARD WIDTH.
C
      IF(DIPWID .LE. STDWID(1))THEN
         DIPWID=STDWID(1)
      ELSEIF(DIPWID .GE. STDWID(NUMWID))THEN
         DIPWID=STDWID(NUMWID)
      ELSE
         DO 1 I=2,NUMWID,1
            DELWID2=0.5*(STDWID(I)-STDWID(I-1))
            IF(DIPWID .LE. STDWID(I))THEN
               IF((DIPWID-STDWID(I-1)) .GT. DELWID2)THEN
                  DIPWID=STDWID(I)
               ELSE
                  DIPWID=STDWID(I-1)
               ENDIF
               GO TO 2
            ENDIF
    1    CONTINUE
    2    CONTINUE
      ENDIF
C
C    CONVERT WIDTH BACK TO UNITS OF "UNITS".
C
      DIPWID=DIPWID/UNIT2IN(1.0,UNITS)
      RETURN
      END
```

```
$PAGESIZE:80
$LINESIZE:132
$MESSAGE:'Compiling GETBEAM'
$SUBTITLE: '    SUBROUTINE GETBEAM'
$PAGE
$NOTRUNCATE
      SUBROUTINE GETBEAM(MXNBEAM,NUMBEAM,AZBEAM,ELBEAM)
C
C   LAST MODIFIED 10/18/88.  BY R. WILSON.
C   ************* 11/01/88.  BY R. WILSON.
C         1)  Changed from using "THBEAM" and "PHIBEAM" (spherical angles)
C             to directly using "AZBEAM" and "ELBEAM".
C   *************  3/09/89.  BY R. WILSON.
C         1)  Added handling of multiple beams.
C   *************  3/14/89.  BY R. WILSON.
C         1)  Added protection from specifying duplicate beam positions when
C             specifying multiple beams.
C   *************  7/20/89.  BY R. WILSON.
C         1) Added trapping of operator data entry errors.
C         2) Previously a typing mistake would terminate the program with a
C            run-time error message and a return to DOS.
C
C   "GETBEAM" prompts for the desired number of main beams and direction for
C   for each of the beams.
C
C   "MXNBEAM" is the maximum number of beams allowed.  It is used as the
C   dimension for "AZBEAM" and "ELBEAM" and as the upper limit for "NUMBEAM".
C   "MXNBEAM" is an input to "GETBEAM".
C
C   "NUMBEAM" is the number of main beams desired.  "GETBEAM" checks to make
C   sure that "NUMBEAM" is at least 1 and no greater than "MXNBEAM".  "NUMBEAM"
C   is an output from "GETBEAM".
C
C   "AZBEAM" and "ELBEAM" are arrays which hold the azimuth and elevation
C   angles, respectively, for each main beam.  The angles are assumed to be in
C   degrees.  The "AZBEAM(I)" and "ELBEAM(I)" are outputs from "GETBEAM".
C
      REAL AZBEAM(MXNBEAM),ELBEAM(MXNBEAM)
      INTEGER NUMBEAM
C
C   Display opening instructions.
C
      WRITE(*,100)
  100 FORMAT(/1X,'When prompted enter the desired number of beams,',
     $  1X,'and the AZIMUTH and ELEVATION'/1X,'angle (in degrees) for',
     $  1X,'each beam.')
C
C   Get desired number of beams.
C
      NUMBEAM=0
      DO WHILE (NUMBEAM.LT.1 .OR. NUMBEAM.GT.MXNBEAM)
         WRITE(*,101)MXNBEAM
  101    FORMAT(/5X,'Enter desired number of beams (1-',I1,'): '\)
         READ(*,*,IOSTAT=IOSTAT)NUMBEAM
         DO WHILE (IOSTAT .NE. 0)
            CALL READERR
            WRITE(*,101)MXNBEAM
            READ(*,*,IOSTAT=IOSTAT)NUMBEAM
         END DO
         IF(NUMBEAM.GE.1 .AND. NUMBEAM.LE.MXNBEAM)EXIT
         CALL BEEP
         WRITE(*,102)MXNBEAM
  102    FORMAT(/1X,'***** ERROR - Number of beams must be between',
     $      1X,'1 and ',I1,'.  Try Again.')
      END DO
C
C   Get desired direction of each beam.
C
      DO 3 IBEAM=1,NUMBEAM,1
```

```
    1     CONTINUE
          WRITE(*,104)IBEAM
  104     FORMAT(5X,'Enter desired position of beam #',I1,
     $      ' (AZ,EL): '\)
          READ(*,*,IOSTAT=IOSTAT)AZBEAM(IBEAM),ELBEAM(IBEAM)
          DO WHILE (IOSTAT .NE. 0)
             CALL READERR
             WRITE(*,104)IBEAM
             READ(*,*,IOSTAT=IOSTAT)AZBEAM(IBEAM),ELBEAM(IBEAM)
          END DO
C
C      Check for duplicate beam positions.
C
          IF(IBEAM .GT. 1)THEN
             DO 2 J=1,IBEAM-1,1
                IF(AZBEAM(IBEAM) .EQ. AZBEAM(J) .AND.
     $            ELBEAM(IBEAM) .EQ. ELBEAM(J))THEN
                   CALL BEEP
                   WRITE(*,105)
  105              FORMAT(/1X,'***** ERROR - You have duplicated a beam',
     $               1X,'position.  Try again.'/)
                   GO TO 1
                ENDIF
    2        CONTINUE
          ENDIF
    3 CONTINUE
      RETURN
      END
```

```
$PAGESIZE:80
$LINESIZE:132
$MESSAGE:'Compiling GETFREQ'
$SUBTITLE: '    SUBROUTINE GETFREQ'
$PAGE
$NOTRUNCATE
      SUBROUTINE GETFREQ(FREQGHZ,FREQ2K0,K0,LAMBDA)
C
C   LAST MODIFIED 8/24/88.   BY R. WILSON.
C   *************  7/20/89.  BY R. WILSON.
C        1) Added trapping of operator data entry errors.
C        2) Previously a typing mistake would terminate the program with a
C           run-time error message and a return to DOS.
C
      PARAMETER(PI=3.1415926535898,TWOPI=2.0*PI)
      REAL FREQGHZ,K0,LAMBDA
C
C   Obtain the source frequency.
C
      WRITE(*,100)
  100 FORMAT(/1X,'Enter the desired source frequency',
     $  1X,'(in GHz): '\)
      READ(*,*,IOSTAT=IOSTAT)FREQGHZ
      DO WHILE (IOSTAT .NE. 0)
         CALL READERR
         WRITE(*,100)
         READ(*,*,IOSTAT=IOSTAT)FREQGHZ
      END DO
      K0=FREQ2K0*FREQGHZ
      LAMBDA=TWOPI/K0
      RETURN
      END
```

```
$PAGESIZE:80
$LINESIZE:132
$MESSAGE:'Compiling GETGEOM'
$SUBTITLE: '    SUBROUTINE GETGEOM'
$PAGE
$NOTRUNCATE
      SUBROUTINE GETGEOM(UNITS,LAMBDA,NUMBEAM,AZBEAM,ELBEAM,PANTILT,
     $  DIPTILT,XSIZE,YSIZE,NXEL,NXELMX,NYEL,NYELMX,XLOCATE,YLOCATE,
     $  DIPWID,DIPLNG,DX,DY,LFILNET,SFILNET,GNDPLN)
C
C    LAST MODIFIED 10/26/88.  BY R. WILSON.
C    ************** 10/27/88.  BY R. WILSON.
C          1) ADDED CODE TO AUTOMATICALLY DETERMINE THE REQUIRED SPACING
C             IN X AND Y TO PREVENT GRATING LOBES.
C          2) ADDED CODE TO AUTOMATICALLY DETERMINE THE REQUIRED NUMBER ARRAY
C             ELEMENTS IN X AND Y.
C          3) ADDED CODE TO AUTOMATICALLY DETERMINE THE WIDTH OF THE DIPOLES.
C             THE NEAREST "STANDARD" WIDTH IS USED.
C          4) ADDED CODE TO AUTOMATICALLY DETERMINE THE ELEMENT LENGTHS WHICH
C             WILL FIT WITHIN THE CALCULATED SPACING.
C    ************** 11/01/88.  BY R. WILSON.
C          1) CHANGED FROM USING "THBEAM" AND "PHIBEAM" (SPHERICAL ANGLES) TO
C             DIRECTLY USING "AZBEAM" AND "ELBEAM".
C    ************** 11/08/88.  BY R. WILSON.
C          1) ADDED HANDLING OF PANEL TILT.
C    **************  3/09/89.  BY R. WILSON.
C          1) ADDED HANDLING OF MULTIPLE BEAMS.
C    **************  4/04/89.  BY R. WILSON.
C          1) ADDED HANDLING OF A GROUND PLANE.
C    **************  4/25/89.  BY R. WILSON.
C          1) MOVED CODE FOR CHOSING NUMBER OF ELEMENTS AND THEIR SPACING TO
C             "GNUMEL".
C          2) MOVED CODE FOR COMPUTING ELEMENT LOCATIONS TO "GLOCATE".
C    **************  5/08/89.  BY R. WILSON.
C          1) CHANGED "REQUIRED ARRAY SIZE TOO BIG" ERROR MESSAGE.
C          2) INSTEAD OF TERMINATING ON THIS ERROR NOW GO BACK AND ASK FOR SIZE.
C    **************  5/16/89.  BY R. WILSON.
C          1) ADDED COMPENSATION FOR NON-AIR SPACER MATERIAL IN THE
C             COMPUTED VALUE OF THE GROUND PLANE DISTANCE.
C          2) TWO DISTANCES ARE NOW COMPUTED, "GDNDPLD", AND "DGNDPLA".
C    **************  6/21/89.  BY R. WILSON.
C          1) CHANGED TO USING AN ARRAY, "GNDPLN", TO HOLD THE TWO GROUND PLANE
C             DISTANCES AND ADDED THE SPACER DIELECTRIC CONSTANT AS THE THIRD
C             VALUE STORED IN "GNDPLN".
C
C    "GETGEOM" PROMPTS THE USER FOR THE DESIRED MAXIMUM SIZE OF THE ARRAY AND
C    THE LOCATION OF THE FILTER NETWORK.  FROM THIS INFORMATION AND THE BEAM
C    DIRECTION INPUT TO "GETGEOM" IT COMPUTES THE NUMBER OF DIPOLES NEEDED IN
C    THE ARRAY, THE ELEMENT SPACING, AN ADJUSTED ARRAY SIZE, AND THE (X,Y)
C    LOCATIONS OF EACH DIPOLE IN THE ARRAY.  THE DIPOLE WIDTHS AND LENGTHS ARE
C    THEN DETERMINED.
C
C    DESCRIPTION OF PARAMETERS
C
C    "UNITS" IS A CHARACTER VARIABLE CONTAINING THE UNITS USED FOR LENGTHS
C    AND POSITION.  IT IS PROMPTED FOR BY "GETUNIT".  IT IS AN INPUT TO
C    "GETGEOM".
C
C    "LAMBDA" IS THE WAVELENTH OF THE SOURCE.  THE UNITS FOR "LAMBDA" (AS FOR
C    ALL LENGTHS IS) "UNITS".  IT IS AN INPUT TO "GETGEOM".
C
C    "NUMBEAM" IS THE NUMBER OF BEAMS FOR THIS DESIGN.  IT IS AN INPUT TO
C    "GETGEOM".
C
C    "AZBEAM" AND "ELBEAM" ARE ARRAYS HOLDING THE AZIMUTH AND ELEVATION ANGLE
C    FOR EACH BEAM IN DEGREES.  THEY ARE INPUTS TO "GETGEOM".
C
C    "PANTILT" IS THE ELEVATION ANGLE OF THE PANEL NORMAL IN DEGREES.  WHEN
C    "PANTILT" IS 0 THE PANEL IS PERPENDICULAR TO THE GROUND.  IT IS AN OUTPUT
```

```
C     FROM "GETGEOM".
C
C     "DIPTILT" IS AN ARRAY CONTAINING A UNIT DIRECTION VECTOR SPECIFYING THE
C     DIRECTION OF THE DIPOLES IN THE APERTURE PLANE.  IT SHOULD BE DIMENSIONED
C     THREE (3) IN THE CALLING PROGRAM.  SINCE THE Z COMPONENT OF "DIPTILT" WILL
C     ALWAYS BE 0 THE ANGLE OF THE DIPOLES, IN DEGREES, WITH RESPECT TO THE
C     APERTURE VERTICAL IS STORED IN "DIPTILT(3)".  "DIPTILT" IS AN OUTPUT FROM
C     "GETGEOM"
C
C     "XSIZE" AND "YSIZE" ARE THE HORIZONTAL AND VERTICAL DIMENSIONS OF THE
C     ARRAY.  THE USER IS PROMPTED FOR THESE DIMENSIONS.  "GETGEOM" THEN
C     DETERMINES THE NUMBER AND SIZES OF THE DIPOLES, CENTERING THE ACTIVE AREA
C     OF THE ARRAY WITHIN THE AREA DEFINED BY "XSIZE" AND "YSIZE".  "XSIZE" AND
C     "YSIZE" ARE ASSUMED TO HAVE BEEN ENTERED IN THE UNITS CONTAINED IN "UNITS".
C     THEY ARE OUTPUTS FROM "GETGEOM".
C
C     "NXEL" AND "NYEL" ARE THE NUMBER OF DIPOLE ELEMENTS IN THE HORIZONTAL AND
C     VERTICAL DIRECTIONS RESPECTIVELY.  THEY ARE DETERMINED BY "GETGEOM" SO AS
C     TO HAVE THE ARRAY FIT WITHIN THE SPECIFED AREA (SPECIFIED BY "XSIZE" AND
C     "YSIZE").  THEY ARE OUTPUTS FROM "GETGEOM".
C
C     "NXELMX" AND "NYELMX" ARE THE UPPER BOUNDS ON "NXEL" AND "NYEL"
C     RESPECTIVELY.  THEY ARE ALSO THE DIMENSION OF THE ARRAYS IN THE CALLING
C     PROGRAM CORRESPONDING TO "XLOCATE" AND "YLOCATE".  THEY ARE INPUTS TO
C     "GETGEOM".
C
C     "XLOCATE" AND "YLOCATE" ARE ONE DIMENSIONAL ARRAYS CONTAINING THE
C     HORIZONTAL AND VERTICAL LOCATIONS OF EACH DIPOLE ELEMENT.  FOR EXAMPLE, THE
C     LOCATION OF THE (2,4)TH ELEMENT IN THE ARRAY IS (XLOCATE(2),YLOCATE(4)).
C     "XLOCATE" AND "YLOCATE" ARE ASSUMED TO HAVE THE UNITS OF "UNITS".  THEY ARE
C     OUTPUTS FROM "GETGEOM".
C
C     "DIPWID" IS THE WIDTH OF THE CONDUCTORS MAKING UP EACH DIPOLE IN THE ARRAY.
C     SEE "GDWIDTH" FOR DETAILS.  "DIPWID" IS AN OUTPUT FROM "GETGEOM".
C
C     "DIPLNG" IS THE LENGTH IN UNITS OF "UNITS" OF EACH DIPOLE IN THE ARRAY.
C     IT IS AN OUTPUT FROM "GETGEOM".
C
C     "DX" AND "DY" ARE THE HORIZONTAL AND VERTICAL SPACING IN UNITS OF "UNITS"
C     BETWEEN THE CENTERS OF THE DIPOLES IN THE ARRAY.  THEY ARE OUTPUTS FROM
C     "GETGEOM".
C
C     "LFILNET" IS A ONE CHARACTER VARIABLE INDICATING THE DESIRED LOCATION OF
C     THE FILTER NETWORK ON THE MYLAR.  SEE "GFILNET" FOR POSSIBLE VALUES AND
C     THEIR DESCRIPTION.  "LFILNET" IS AN OUTPUT FROM "GETGEOM".
C
C     "SFILNET" IS THE PERPENDICULAR DISTANCE, IN UNITS OF "UNITS", FROM THE
C     "LFILNET" EDGE OF THE MYLAR SPECIFYING THE SIZE OF THE SPACE REQUIRED FOR
C     THE FILTER NETWORK.  "SFILNET" IS AN OUTPUT FROM "GETGEOM".
C
C     "GNDPLN" IS A THREE ELEMENT REAL ARRAY HOLDING PARAMETERS FOR DESIGN OF A
C     GROUND PLANE.  IF A GROUND PLANE IS TO BE USED, "GNDPLN(1)" IS THE
C     ELECTRICAL DISTANCE TO THE GROUND PLANE THROUGH THE DIELECTRIC OF THE
C     SPACER MATERIAL.  "GNDPLN(2)" IS THE PHYSICAL DISTANCE TO THE GROUND PLANE.
C     UNITS FOR "GNDPLN(1)" AND "GNDPLN(2)" ARE THE SAME AS THE UNITS FOR
C     "LAMBDA".  "GNDPLN(3)" STORES THE DIELECTRIC CONSTANT OF THE SPACER
C     MATERIAL BETWEEN THE ARRAY PANEL AND THE GROUND PLANE.  IF A GROUND PLANE
C     IS NOT DESIRED "GNDPLN(1)" AND "GNDPLN(2)" ARE SET LESS THAN ZERO,
C     "GNDPLN(3)" WILL BE UNDEFINED.  "GNDPLN(.)" ARE OUTPUTS FROM "GETGEOM".
C
        REAL XLOCATE(NXELMX),YLOCATE(NYELMX)
        REAL AZBEAM(NUMBEAM),ELBEAM(NUMBEAM)
        REAL DIPTILT(3),GNDPLN(3),LAMBDA
        LOGICAL YESNOTF
        CHARACTER UNITS*2,LFILNET*1
C
C     OBTAIN THE DESIRED PANEL ELEVATION TILT.
C
        CALL GPTILT(PANTILT)
```

```
C
C   OBTAIN THE DESIRED DIPOLE TILT ANGLES, THE "TILT OF THE LINEAR
C   POLARIZATION".
C
      CALL GDTILT(DIPTILT)
C
C  GET THE DESIRED LOCATION OF THE MATCHING NETWORK.
C
      CALL GFILNET(LAMBDA,DIPTILT(3),LFILNET,SFILNET)
C
C   DETERMINE THE CONDUCTOR WIDTH, "DIPWID", TO BE USED FOR THE DIPOLES.
C
      CALL GDWIDTH(UNITS,LAMBDA,DIPWID)
C
C   DETERMINE IF A GROUND PLANE IS DESIRED AND, IF SO, ITS LOCATION.
C
      CALL GGNDPLN(LAMBDA,NUMBEAM,AZBEAM,ELBEAM,PANTILT,GNDPLN)
C
C  OBTAIN THE DESIRED MAXIMUM SIZE OF THE ARRAY.
C
    1 continue
      CALL GSIZE(UNITS,LFILNET,SFILNET,XSIZE,XACTIVE,YSIZE,YACTIVE)
C
C   COMPUTE THE NUMBER OF ELEMENTS IN THE ARRAY AND THE ELEMENT SPACING.
C
      CALL GNUMEL(LAMBDA,NUMBEAM,AZBEAM,ELBEAM,PANTILT,XACTIVE,
     $  YACTIVE,NXEL,NYEL,DX,DY)
C
C   CHECK IF NUMBER OF ELEMENTS IS ALLOWED.
C
      IF(NXEL.GT.NXELMX .OR. NYEL.GT.NYELMX)THEN
         CALL BEEP
         WRITE(*,100)NXEL,NYEL,NXELMX,NYELMX
  100    FORMAT(/1X,'***** ERROR - The required array is too big.'/
     $      7X,'To meet your specifications the array requires ',I3,
     $      1X,'elements horizontally'/7X,'and ',I3,' elements',
     $      1X,'vertically.  The maximum size antenna currently allowed'/
     $      7X,'is ',I3,' by ',I3,'.  To continue you must decrease the',
     $      1X,'size of the array'/7X,'panel.'/)
         GO TO 1
      ENDIF
C
      write(*,*)
      write(*,*)'The actual array size is ',XSIZE,' by ',YSIZE
      write(*,*)'The active array size is ',xactive,' by ',yactive
      write(*,101)nxel,nyel
  101 FORMAT(1X,'There are',I3,' elements HORIZONTALLY and ',I3,
     $  1X,'elements VERTICALLY')
      write(*,*)'  HORIZONTAL spacing=',dx,' VERTICAL spacing=',dy
      if(.not.(YesNoTF('Is this OK')))go to 1
C
C   LOAD LOCATION ARRAYS "XLOCATE" AND "YLOCATE".
C
      CALL GLOCATE(NXEL,NYEL,DX,DY,XACTIVE,YACTIVE,LFILNET,
     $  SFILNET,XLOCATE,YLOCATE)
C
C   DETERMINE THE LENGTHS OF THE DIPOLES IN THE ARRAY.
C
      CALL GDLNGTH(LAMBDA,NXEL,NYEL,DX,DY,DIPWID,DIPTILT,XACTIVE,
     $  YACTIVE,DIPLNG)
      RETURN
      END
```

```
$PAGESIZE:80
$LINESIZE:132
$MESSAGE:'Compiling GETUNIT'
$SUBTITLE: '    SUBROUTINE GETUNIT'
$PAGE
$NOTRUNCATE
      SUBROUTINE GETUNIT(UNITS,FREQ2K0)
C
C   LAST MODIFIED  8/24/88.  BY R. WILSON.
C   **************  8/03/89.  BY R. WILSON.
C        1) Changed selection method from typing letters in parenthesis to
C           numbers alongside the choice.
C        2) Re-ordered the choices so that English units come first, then
C           metric units.  The smallest units in each group come first then
C           the largest.  This was done at the suggestion of Joe Zimmerman,
C           et. al. at the meeting the end of July 1989.
C
      PARAMETER(PI=3.1415926535898,TWOPI=2.0*PI)
      CHARACTER ICHOICE*1,UNITS*2
C
C   Obtain the units desired for the linear dimensions of the array,"UNITS".
C   Initialize the frequency to wavenumber conversion parameter, "FREQ2K0".
C
    1 CONTINUE
      WRITE(*,100)
  100 FORMAT(/1X,'Please specify the units you want to use for',
     $ 1X,'the array size and element'/1X,'position from one of the',
     $ 1X,'following:'//20X,'0 - inches,'/20X,'1 - feet,'/
     $ 20X,'2 - centimeters,'/20X,'3 - meters.'//
     $ 1X,'Make your selection by entering the number corresponding',
     $ 1X,'to your choice.'//1X,'I want to use: '\)
      READ(*,'(A1)',END=1)ICHOICE
      SELECT CASE (ICHOICE)
         CASE ('0')
            UNITS(1:2)='in'
            FREQ2K0=TWOPI*2.54/29.97925
         CASE ('1')
            UNITS(1:2)='ft'
            FREQ2K0=TWOPI*30.48/29.97925
         CASE ('2')
            UNITS(1:2)='cm'
            FREQ2K0=TWOPI/29.97925
         CASE ('3')
            UNITS(1:2)='m'
            FREQ2K0=TWOPI/0.2997925
         CASE DEFAULT
            CALL BEEP
            WRITE(*,101)
  101       FORMAT(/1X,'***** ERROR - incorrect units choice.  Re-read',
     $         1X,'the instructions and try again.'/)
            GO TO 1
      END SELECT
      RETURN
      END
```

```
$PAGESIZE:80
$LINESIZE:132
$MESSAGE:'Compiling GFILNET'
$SUBTITLE: '    SUBROUTINE GFILNET'
$PAGE
$NOTRUNCATE
      SUBROUTINE GFILNET(LAMBDA,DIPTILT,LFILNET,SFILNET)
C
C   LAST MODIFIED 11/17/88.  BY R. WILSON.
C   **************  4/04/89.  BY R. WILSON.
C        1) Changed prompt for filter network location so that only "L" and
C           "R" are allowed for 0 < ABS( "DIPTILT)" < 45, and "T" and "B" only
C           are allowed for 45 < ABS("DIPTILT") < 90.  "DIPTILT" can be any
C           angle, however, before the above criterion is applied "DIPTILT"
C           adjusted so that 0 < ABS("DIPTILT") < 90.  Note that "DIPTILT" is
C           not RETURNED changed by the routine.
C
C   "GFILNET" first prompts for the desired location of the filter network on
C   the array panel.  It then initializes the size of the space on the array
C   panel to be left for the filter network. "GFILNET" stands for Get_FILter_
C   NETwork.
C
C   "LAMBDA" is the wavelength of the radiation.  It is an input to "GFILNET".
C
C   "DIPTILT" is the tilt angle of the dipoles in degrees.  It is used to
C   determine the allowed locations of the filter network.  Unlike in other
C   routines, where "DIPTILT" is an array, here "DIPTILT" is a single number.
C   in fact it is equal to "DIPTILT(3)" in the other routines.  "DIPTILT" is an
C   input to "GFILNET".
C
C   The location is returned in the character variable "LFILNET".  Valid values
C   for "LFILNET" are: T - FOR THE TOP EDGE
C                      B - FOR THE BOTTOM EDGE
C                      L - FOR THE LEFT EDGE
C                      R - FOR THE RIGHT EDGE
C   "LFILNET" is returned upper case, however, the user can respond with either
C   upper or lower case.  "GFILNET" does the conversion to upper case.
C   "LFILNET" is an output from "GFILNET".
C
C   The space for the filter network is returned in "SFILNET".  Whether this
C   dimension will be along the X direction or the Y direction will depend on
C   the desired location of the filter network.  "SFILNET" is an output from
C   "GFILNET".
C
      REAL LAMBDA,SFILNET
      CHARACTER LFILNET*1
C
C   Initialize size of the space to be left for the filter network, "SFILNET".
C
      SFILNET=0.5*LAMBDA
C
C   Adjust the tilt angle, if necessary, so that 0 < DIPTLT < 90.0
C
      DIPTLT=ABS(DIPTILT)
      IF(DIPTLT .GT. 90.0)DIPTLT=180.0-DIPTLT
C
C   Get the desired location of the filter network, "LFILNET".
C
      IF(DIPTLT .LT. 45.0)THEN
    1   CONTINUE
        WRITE(*,100)
  100   FORMAT(/1X,'Space must be left for the filter network along',
     $    1X,'the'//20X,'(L)eft edge, or'/20X,'(R)ight edge'//
     $    1X,'of the array panel.  Indicate your choice by typing the',
     $    1X,'corresponding letter'/1X,'shown in parenthesis.',
     $    1X,'I want: '\)
        READ(*,'(A1)')LFILNET
        CALL UPPER(LFILNET,1)
        IF(LFILNET.NE.'L' .AND. LFILNET.NE.'R')THEN
```

```
                CALL BEEP
                WRITE(*,101)
 101            FORMAT(/1X,'***** ERROR - incorrect choice.  Re-read the',
     $            1X,'instructions and try again.'/)
                GO TO 1
             ENDIF
          ELSE
   2         CONTINUE
             WRITE(*,102)
 102         FORMAT(/1X,'Space must be left for the filter network along',
     $         1X,'the'//20X,'(T)op edge, or'/20X,'(B)ottom edge'//
     $         1X,'of the array panel.  Indicate your choice by typing the',
     $         1X,'corresponding letter'/1X,'shown in parenthesis.',
     $         1X,'I want: '\)
             READ(*,'(A1)')LFILNET
             CALL UPPER(LFILNET,1)
             IF(LFILNET.NE.'T' .AND. LFILNET.NE.'B')THEN
                CALL BEEP
                WRITE(*,101)
                GO TO 2
             ENDIF
          ENDIF
          RETURN
          END
```

```
$PAGESIZE:80
$LINESIZE:132
$MESSAGE:'Compiling GGNDPLN'
$SUBTITLE: '    SUBROUTINE GGNDPLN'
$PAGE
$NOTRUNCATE
      SUBROUTINE GGNDPLN(LAMBDA,NUMBEAM,AZBEAM,ELBEAM,PANTILT,GNDPLN)
C
C   LAST MODIFIED  4/04/89.  BY R. WILSON.
C   *************  5/16/89.  BY R. WILSON.
C        1) Added compensation for non-air spacer material in the
C           computed value of the ground plane distance.
C        2) Two distances are now computed, "DGNDPLD", and "DGNDPLA".
C   *************  6/12/89.  BY R. WILSON.
C        1) Changed references to mylar in prompts to array substrate.
C   *************  6/21/89.  BY R. WILSON.
C        1) Changed to using an array, "GNDPLN", to hold the two ground plane
C           distances and added the spacer dielectric constant as the third
C           value stored in "GNDPLN".
C   *************  7/20/89.  BY R. WILSON.
C        1) Added trapping of operator data entry errors.
C        2) Previously a typing mistake would terminate the program with a
C           run-time error message and a return to DOS.
C   *************  9/15/89.  BY R. WILSON.
C        1) Added listing for dielectric constant of polystyrene to the prompt
C           as suggested by Joe Zimmerman.
C
C   "GGNDPLN" determines if a ground plane is desired and, if so, computes
C   the distance to the ground plane.
C
C   "LAMBDA" is the wavelength of the source.  The units for "LAMBDA" are
C   the units for "GNDPLN(1)" and "GNDPLN(2)". It is an input to "GGNDPLN".
C
C   "NUMBEAM" is the nubmer of beams for this design.  It is an input to
C   "GGNDPLN".
C
C   "AZBEAM" and "ELBEAM" are arrays holding the AZIMUTH and ELEVATION angle
C   for each beam in degrees.  They are inputs to "GGNDPLN".
C
C   "PANTILT" is the ELEVATION angle of the panel normal in degrees.  When
C   "PANTILT" is 0 the panel is perpendicular to the ground.  It is an input
C   to "GGNDPLN".
C
C   "GNDPLN" is a three element real array holding parameters for design of a
C   ground plane.  If a ground plane is to be used, "GNDPLN(1)" is the
C   electrical distance to the ground plane through the dielectric of the
C   spacer material.  "GNDPLN(2)" is the physical distance to the ground plane.
C   Units for "GNDPLN(1)" and "GNDPLN(2)" are the same as the units for
C   "LAMBDA".  "GNDPLN(3)" stores the dielectric constant of the spacer
C   material between the array panel and the ground plane.  If a ground plane
C   is not desired "GNDPLN(1)" and "GNDPLN(2)" are set less than zero,
C   "GNDPLN(3)" will be undefined   "GNDPLN(.)" are outputs from "GGNDPLN".
C
      REAL AZBEAM(NUMBEAM),ELBEAM(NUMBEAM),KBEAM(3),GNDPLN(3)
      REAL LAMBDA,LAMBDA4,PANTILT
      LOGICAL UGNDPLN,YESNOTF
C
C   Determine if a ground plane is desired.
C
      WRITE(*,*)
      UGNDPLN=YESNOTF('Do you want to use a ground plane')
C
C   Compute the distance to the ground plane.
C
      GNDPLN(1)=-1.0
      GNDPLN(2)=-1.0
      IF(UGNDPLN)THEN
         WRITE(*,100)
  100    FORMAT(/5X,'Please enter the dielectric constant of the',
```

```
     $      1X,'SPACER material'/5X,'between the array substrate and',
     $      1X,'ground plane'/5X,'(typically 1.033 for polystyrene): '\)
          READ(*,*,IOSTAT=IOSTAT)GNDPLN(3)
          DO WHILE (IOSTAT .NE. 0)
             CALL READERR
             WRITE(*,100)
             READ(*,*,IOSTAT=IOSTAT)GNDPLN(3)
          END DO
          GNDPLN(1)=0.0
          GNDPLN(2)=0.0
          LAMBDA4=LAMBDA/4.0
          DO 1 IBEAM=1,NUMBEAM,1
             CALL AZEL2K(AZBEAM(IBEAM),ELBEAM(IBEAM),PANTILT,KBEAM)
             GNDPLN(1)=GNDPLN(1)+LAMBDA4/KBEAM(3)
             GNDPLN(2)=GNDPLN(2)+LAMBDA4/SQRT(GNDPLN(3)+KBEAM(3)**2-1)
    1     CONTINUE
          GNDPLN(1)=GNDPLN(1)/NUMBEAM
          GNDPLN(2)=GNDPLN(2)/NUMBEAM
       ENDIF
       RETURN
       END
```

```
$PAGESIZE:80
$LINESIZE:132
$MESSAGE:'Compiling GLOCATE'
$SUBTITLE: '     SUBROUTINE GLOCATE'
$PAGE
$NOTRUNCATE
      SUBROUTINE GLOCATE(NXEL,NYEL,DX,DY,XACTIVE,YACTIVE,LFILNET,
     $  SFILNET,XLOCATE,YLOCATE)
C
C   LAST MODIFIED 4/26/89.  BY R. WILSON.
C
C   "GLOCATE" LOADS THE HORIZONTAL AND VERTICAL (X AND Y) COORDINATES FOR
C   THE CENTERS OF EACH DIPOLE ELEMENT IN A PLANAR ARRAY.
C
C   "NXEL" AND "NYEL" ARE THE NUMBER OF DIPOLE ELEMENTS IN THE HORIZONTAL AND
C   VERTICAL DIRECTIONS RESPECTIVELY.  THEY ARE INPUT TO "GLOCATE".
C
C   "DX" AND "DY" ARE THE HORIZONTAL AND VERTICAL SPACING BETWEEN THE CENTERS
C   OF THE DIPOLES IN THE ARRAY.  THEIR DIMENSIONS SHOULD BE CONSISTENT WITH
C   THE DIMENSIONS OF "XACTIVE", "YACTIVE", AND "SFILNET".  THEY ARE INPUTS TO
C   "GLOCATE".
C
C   "XACTIVE" AND "YACTIVE" ARE THE HORIZONTAL AND VERTICAL DIMENSIONS,
C   RESPECTIVELY, OF THE ACTIVE AREA OF THE ARRAY PANEL.  THAT IS, THE AREA OF
C   THE PANEL WHICH ACTUALLY CONTAINS THE DIPOLES.  THEIR DIMENSIONS SHOULD BE
C   CONSISTENT WITH THE DIMENSIONS OF "DX", "DY", AND "SFILNET".  THEY ARE
C   INPUTS TO "GLOCATE".
C
C   "LFILNET" IS A ONE CHARACTER VARIABLE INDICATING THE DESIRED LOCATION OF
C   THE FILTER NETWORK ON THE MYLAR.  SEE "GFILNET" FOR POSSIBLE VALUES AND
C   THEIR DESCRIPTION.  "LFILNET" IS AN INPUT TO "GLOCATE".
C
C   "SFILNET" IS THE PERPENDICULAR DISTANCE FROM THE "LFILNET" EDGE OF THE
C   MYLAR SPECIFYING THE SIZE OF THE SPACE REQUIRED FOR THE FILTER NETWORK.
C   ITS DIMENSION SHOULD BE CONSISTENT WITH THE DIMENSIONS OF "DX", "DY",
C   "XACTIVE", AND "YACTIVE".  "SFILNET" IS AN INPUT TO "GLOCATE".
C
C   "XLOCATE" AND "YLOCATE" ARE ONE DIMENSIONAL ARRAYS CONTAINING THE
C   HORIZONTAL AND VERTICAL LOCATIONS OF EACH DIPOLE ELEMENT.  FOR EXAMPLE, THE
C   LOCATION OF THE (2,4)TH ELEMENT IN THE ARRAY IS (XLOCATE(2),YLOCATE(4)).
C   THE CENTERS ARE CHOSEN SO AS TO CENTER THE ELEMENTS WITHIN THE AREA DEFINED
C   BY "XACTIVE" AND "YACTIVE".  THEIR DIMENSIONS ARE ASSUMED TO BE THOSE OF
C   "DX", "DY", "XACTIVE", "YACTIVE", AND "SFILNET".  "XLOCATE" AND "YLOCATE"
C   ARE OUTPUTS FROM "GLOCATE".
C
      REAL XLOCATE(NXEL),YLOCATE(NYEL)
      CHARACTER*1 LFILNET
C
C   LOAD THE LOCATION ARRAYS "XLOCATE" AND "YLOCATE".
C
      XLOCATE(1)=(XACTIVE-(NXEL-1)*DX)/2.0
      IF(LFILNET .EQ. 'L')XLOCATE(1)=XLOCATE(1)+SFILNET
      YLOCATE(1)=(YACTIVE-(NYEL-1)*DY)/2.0
      IF(LFILNET .EQ. 'B')YLOCATE(1)=YLOCATE(1)+SFILNET
      DO 1 I=1,NXEL,1
         XLOCATE(I)=(I-1)*DX+XLOCATE(1)
    1 CONTINUE
      DO 2 I=1,NYEL,1
         YLOCATE(I)=(I-1)*DY+YLOCATE(1)
    2 CONTINUE
      RETURN
      END
```

```
$PAGESIZE:80
$LINESIZE:132
$MESSAGE:'Compiling GNUMEL'
$SUBTITLE: '    SUBROUTINE GNUMEL'
$PAGE
$NOTRUNCATE
      SUBROUTINE GNUMEL(LAMBDA,NUMBEAM,AZBEAM,ELBEAM,PANTILT,XACTIVE,
     $  YACTIVE,NXEL,NYEL,DX,DY)
C
C   LAST MODIFIED 4/26/89.  BY R. WILSON.
C
C   "GNUMEL" DETERMINES THE NUMBER OF ELEMENTS IN AN ARRAY AND THEIR SPACING
C   SUCH THAT GRATING LOBES ARE PREVENTED.
C
C   "LAMBDA" IS THE WAVELENGTH OF THE SOURCE.  THE DIMENSIONS FOR "LAMBDA"
C   DETERMINE THE DIMENSIONS ASSUMED FOR "DX" AND "DY".  "LAMBDA" IS AN
C   INPUT TO "GNUMEL".
C
C   "NUMBEAM" IS THE NUMBER OF BEAMS FOR THIS DESIGN.  IT IS AN INPUT TO
C   "GNUMEL".
C
C   "AZBEAM" AND "ELBEAM" ARE ARRAY HOLDING THE AZIMUTH AND ELEVATION ANGLE
C   FOR EACH BEAM IN DEGREES.  THEY ARE INPUTS TO "GNUMEL".
C
C   "PANTILT" IS THE ELEVATION ANGLE OF THE PANEL NORMAL IN DEGREES.  WHEN
C   "PANTILT" IS 0 THE PANEL IS PERPENDICULAR TO THE GROUND.  IT IS AN INPUT
C   TO "GNUMEL".
C
C   "XACTIVE" AND "YACTIVE" ARE THE HORIZONTAL AND VERTICAL DIMENSIONS,
C   RESPECTIVELY, OF THE ACTIVE AREA OF THE ARRAY PANEL.  THAT IS, THE AREA OF
C   PANEL WHICH ACTUALLY CONTAINS THE DIPOLES.  THEIR DIMENSIONS SHOULD BE THE
C   SAME AS FOR "LAMBDA".  "XACTIVE" AND "YACTIVE" ARE INPUTS TO "GNUMEL".
C
C   "NXEL" AND "NYEL" ARE THE NUMBER OF DIPOLE ELEMENTS IN THE HORIZONTAL AND
C   VERTICAL DIRECTIONS, RESPECTIVELY.  THEY ARE CHOSEN BY "GNUMEL" TO PREVENT
C   GRATING LOBES.  THEY ARE OUTPUTS FROM "GNUMEL".
C
C   "DX" AND "DY" ARE THE HORIZONTAL AND VERTICAL SPACING, RESPECTIVELY,
C   BETWEEN THE CENTERS OF THE DIPOLES IN THE ARRAY.  THEIR DIMENSIONS ARE
C   ASSUMED TO BE THE SAME AS THE DIMENSIONS OF "LAMBDA".  THEY ARE CHOSEN
C   BY "GNUMEL" TO PREVENT GRATING LOBES.  THEY ARE OUTPUTS FROM "GNUMEL.
C
      REAL AZBEAM(NUMBEAM),ELBEAM(NUMBEAM)
      REAL DIPTILT(3),K(3),LAMBDA,KX,KY
C
C   COMPUTE THE NUMBER OF ELEMENTS IN THE ARRAY AND THE ELEMENT SPACING.
C
      CALL AZEL2K(AZBEAM(1),ELBEAM(1),PANTILT,K)
      KX=ABS(K(1))
      KY=ABS(K(2))
      NXEL=INT(XACTIVE*(1.0+KX)/LAMBDA+1.0)
      DX=(1.0-1.0/NXEL)*LAMBDA/(1.0+KX)
      NYEL=INT(YACTIVE*(1.0+KY)/LAMBDA+1.0)
      DY=(1.0-1.0/NYEL)*LAMBDA/(1.0+KY)
      IF(NUMBEAM .GT. 1)THEN
         DO 1 IBEAM=2,NUMBEAM,1
            CALL AZEL2K(AZBEAM(IBEAM),ELBEAM(IBEAM),PANTILT,K)
            KX=ABS(K(1))
            KY=ABS(K(2))
            NEL=INT(XACTIVE*(1.0+KX)/LAMBDA+1.0)
            DEL=(1.0-1.0/NEL)*LAMBDA/(1.0+KX)
            IF(DEL .LT. DX)THEN
               DX=DEL
               NXEL=NEL
            ENDIF
            NEL=INT(YACTIVE*(1.0+KY)/LAMBDA+1.0)
            DEL=(1.0-1.0/NEL)*LAMBDA/(1.0+KY)
            IF(DEL .LT. DY)THEN
               DY=DEL
```

```
                  NYEL=NEL
               ENDIF
    1       CONTINUE
         ENDIF
         RETURN
         END
```

```
$PAGESIZE:80
$LINESIZE:132
$MESSAGE:'Compiling GPTILT'
$SUBTITLE: '    SUBROUTINE GPTILT'
$PAGE
$NOTRUNCATE
      SUBROUTINE GPTILT(PANTILT)
C
C   LAST MODIFIED 11/17/88.  BY R. WILSON.
C   *************  7/20/89.  BY R. WILSON.
C        1) Added trapping of operator data entry errors.
C        2) Previously a typing mistake would terminate the program with a
C           run-time error message and a return to DOS.
C
      REAL PANTILT
C
C   "GPTILT" prompts for the desired panel tilt in degrees and returns it in
C   "PANTILT".  "GPTILT"= GetPanelTILT.
C
      WRITE(*,100)
  100 FORMAT(/1X,'The panel containing the array may be tilted in',
     $  1X,'elevation.  At the prompt'/1X,'specify the desired panel',
     $  1X,'tilt by entering the elevation angle of the panel'/
     $  1X,'normal.'//5X,'Enter desired panel tilt (in degrees): '\)
      READ(*,*,IOSTAT=IOSTAT)PANTILT
      DO WHILE (IOSTAT .NE. 0)
         CALL READERR
         WRITE(*,100)
         READ(*,*,IOSTAT=IOSTAT)PANTILT
      END DO
      RETURN
      END
```

```
$PAGESIZE:80
$LINESIZE:132
$MESSAGE:'Compiling GRAFHRD'
$SUBTITLE: '    SUBROUTINE GRAFHRD'
$PAGE
$NOTRUNCATE
      SUBROUTINE GRAFHRD
C
C   LAST MODIFIED 3/28/89.  BY R. WILSON.
C   *************  7/20/89.  BY R. WILSON.
C        1) Added trapping of operator data entry errors.
C        2) Previously a typing mistake would terminate the program with a
C           run-time error message and a return to DOS.
C
C   "GRAFHRD" prompts for the desired graphics output device and initializes
C   "IOPORT" and "MODEL" for use with "PLOTS".
C
      CHARACTER*1 ICHOICE
      COMMON/PLOT88C/IOPORT,MODEL,FACTR
C
C   Get the desired hardcopy device.
C
    1 CONTINUE
      WRITE(*,100)
  100 FORMAT(/1X,'Graphics can be displayed on the following devices:'//
     $  20X,'0 - an EGA monochrome display,'/
     $  20X,'1 - an IBM CGA display,'/
     $  20X,'2 - an HP 7470A plotter, or'/
     $  20X,'3 - an HP LaserJet printer.'//1X,'Please enter the number',
     $  1X,'corresponding to your choice: '\)
      READ(*,'(A1)',END=1)ICHOICE
      SELECT CASE (ICHOICE)
         CASE ('0')
            IOPORT=96
            MODEL=96
         CASE ('1')
            IOPORT=99
            MODEL=99
         CASE ('2')
            IOPORT=9600
            MODEL=20
         CASE('3')
            IOPORT=0
            MODEL=62
         CASE DEFAULT
            CALL BEEP
            WRITE(*,101)
  101       FORMAT(/1X,'***** ERROR - incorrect choice.  Try again.'/)
            GO TO 1
      END SELECT
      RETURN
      END
```

```
$PAGESIZE:80
$LINESIZE:132
$MESSAGE:'Compiling GSIZE'
$SUBTITLE: '    SUBROUTINE GSIZE'
$PAGE
$NOTRUNCATE
      SUBROUTINE GSIZE(UNITS,LFILNET,SFILNET,XSIZE,XACTIVE,
     $  YSIZE,YACTIVE)
C
C   LAST MODIFIED 11/17/88.  BY R. WILSON.
C   ************* 11/29/88.  BY R. WILSON.
C        1) Moved computation of "XACTIVE" and "YACTIVE" here.
C        2)  Added protection against specifying too small a panel size.
C   *************  7/20/89.  BY R. WILSON.
C        1) Added trapping of operator data entry errors.
C        2) Previously a typing mistake would terminate the program with a
C           run-time error message and a return to DOS.
C
      REAL XSIZE,YSIZE
      CHARACTER UNITS*2,LFILNET*1
C
C   "GSIZE" prompts for the desired horizontal and vertical dimensions of the
C   array panel.
C
C   "UNITS" is a two character variable which contains the units currently in
C   use for linear dimensions.  See "GETUNIT" for allowed values.  "UNITS" is
C   an input to "GSIZE".
C
C   "LFILNET" Is a one character variable which indicates the desired location
C   of the filter network.  See "GFILNET" for allowed values.  "LFILNET" is an
C   input to "GSIZE".
C
C   "SFILNET" is the space in units of "UNITS" on the array "PANEL" for the
C   filter network.  "SFILNET" is computed in "GFILNET".  It is an input to
C   "GSIZE".
C
C   The horizontal dimension is returned in "XSIZE" and the vertical dimension
C   is returned in "YSIZE".  "XSIZE" and "YSIZE" include the space to be left
C   for the filter network.  They are in units of "units" and are outputs from
C   "GSIZE".
C
C   "XACTIVE" and "YACTIVE" are the horizontal and vertical dimensions,
C   respectively, of the active area of the array panel.  That is, the area of
C   the panel which actually contains the dipoles.  They are outputs from
C   "GSIZE".
C
      WRITE(*,100)
  100 FORMAT(/1X,'At the prompt enter the desired dimensions for the',
     $  1X,'array panel, including the '/1X,'filter network.  The',
     $  1X,'array will be centered within this area after leaving'/
     $  1X,'space for the filter network.')
      XACTIVE=0.0
      YACTIVE=0.0
    1 CONTINUE
      IF(XACTIVE .LE. 0.0)THEN
         WRITE(*,101)UNITS
  101    FORMAT(/5X,'Enter desired HORIZONTAL dimension (in ',
     $      A2,'.): '\)
         READ(*,*,IOSTAT=IOSTAT)XSIZE
         DO WHILE (IOSTAT .NE. 0)
            CALL READERR
            WRITE(*,101)UNITS
            READ(*,*,IOSTAT=IOSTAT)XSIZE
         END DO
      ENDIF
      IF(YACTIVE .LE. 0.0)THEN
         WRITE(*,102)UNITS
  102    FORMAT(5X,'Enter desired VERTICAL dimension (in ',A2,'.): '\)
         READ(*,*,IOSTAT=IOSTAT)YSIZE
```

```
         DO WHILE (IOSTAT .NE. 0)
            CALL READERR
            WRITE(*,102)UNITS
            READ(*,*,IOSTAT=IOSTAT)YSIZE
         END DO
      ENDIF
      IF(LFILNET.EQ.'T' .OR. LFILNET.EQ.'B')THEN
         XACTIVE=XSIZE
         YACTIVE=YSIZE-SFILNET
         IF(YACTIVE .LE. 0.0)THEN
            CALL BEEP
            WRITE(*,103)SFILNET,UNITS
103         FORMAT(/1X,'***** ERROR - your specified VERTICAL',
     $        1X,'dimension is too small.  The filter'/7X,'network',
     $        1X,'requires at least ',F8.3,1X,A2,'.  Try again.'/)
            GO TO 1
         ENDIF
      ELSE
         XACTIVE=XSIZE-SFILNET
         YACTIVE=YSIZE
         IF(XACTIVE .LE. 0.0)THEN
            CALL BEEP
            WRITE(*,104)SFILNET,UNITS
104         FORMAT(/1X,'***** ERROR - your specified HORIZONTAL',
     $        1X,'dimension is too small.  The filter'/7X,'network',
     $        1X,'requires at least ',F8.3,1X,A2,'.  Try again.'/)
            GO TO 1
         ENDIF
      ENDIF
      RETURN
      END
```

```
$PAGESIZE:80
$LINESIZE:132
$MESSAGE:'Compiling GSUBSTR'
$SUBTITLE: '    SUBROUTINE GSUBSTR'
$PAGE
$NOTRUNCATE
      SUBROUTINE GSUBSTR(UNITS,ERSUB,THSUB)
C
C   LAST MODIFIED  5/08/89.  BY R. WILSON.
C   *************  6/12/89.  BY R. WILSON.
C        1) Changed name of subroutine from "GMYLAR" to "GSUBSTR".
C        2) This reflects the change in the prompts from references to mylar
C           To references to array substrate.
C   *************  7/20/89.  BY R. WILSON.
C        1) Added trapping of operator data entry errors.
C        2) Previously a typing mistake would terminate the program with a
C           run-time error message and a return to DOS.
C   *************  9/15/89.  BY R. WILSON.
C        1) Added display of the dielectric constant for Teflon glass and Mylar
C           to the prompts at the suggestion of Joe Zimmerman.
C
C   "GSUBSTR" prompts for the array substrate dielectric constant, "ERSUB",
C   and thickness, "THSUB".  The thickness is assumed to be in units of
C   "UNITS".
C
      CHARACTER UNITS*2
C
C   Get the dielectric constant and thickness of the array substrate.
C
      WRITE(*,100)
  100 FORMAT(/1X,'In order to design the filter network the',
     $  1X,'dielectric constant and thickness'/1X,'of the SUBSTRATE',
     $  1X,'must be specified.  Please enter these values at the',
     $  1X,'prompts.'/1X,'(Teflon glass= 2.76  Mylar=2.25)')
      WRITE(*,101)
  101 FORMAT(/5X,'Enter substrate DIELECTRIC CONSTANT: '\)
      READ(*,*,IOSTAT=IOSTAT)ERSUB
      DO WHILE (IOSTAT .NE. 0)
         CALL READERR
         WRITE(*,101)
         READ(*,*,IOSTAT=IOSTAT)ERSUB
      END DO
      WRITE(*,102)' ',UNITS
  102 FORMAT(A,4X,'Enter substrate THICKNESS (in ',A2,'.): '\)
      READ(*,*,IOSTAT=IOSTAT)THSUB
      DO WHILE (IOSTAT .NE. 0)
         CALL READERR
         WRITE(*,102)'0',UNITS
         READ(*,*,IOSTAT=IOSTAT)THSUB
      END DO
      RETURN
      END
```

```fortran
$PAGESIZE:80
$LINESIZE:132
$MESSAGE:'Compiling KING1'
$SUBTITLE: '    SUBROUTINE KING1'
$PAGE
$NOTRUNCATE
      SUBROUTINE KING1(D,H,RL1,RL2,ZR,ZX)
C
C     Written by Mike Guler
C     LAST MODIFIED 02-10-89
C
      PARAMETER (PI=3.1415926535898,B=2.*PI)
      REAL U(18),SI(18),CI(18)
      HM1=H-RL1
      HP1=H+RL1
      HP2=H+RL2
      HPT2=H+2.*RL2
      HM12=H-RL1+RL2
      HP12=H+RL1+RL2
      HM1T2=HPT2-RL1
      HP1T2=HPT2+RL1
C
C
      CALL SQROOT(D,HM1,U(1),U(2))
      CALL SQROOT(D,HP1,U(4),U(3))
      CALL SQROOT(D,HM12,U(5),U(6))
      CALL SQROOT(D,HP12,U(8),U(7))
      CALL SQROOT(D,HM1T2,U(9),U(10))
      CALL SQROOT(D,HP1T2,U(12),U(11))
      CALL SQROOT(D,H,U(14),U(13))
      CALL SQROOT(D,HP2,U(16),U(15))
      CALL SQROOT(D,HPT2,U(18),U(17))
C
C
      DO 10 I=1,18
        CALL SICI(U(I),SI(I),CI(I))
   10 CONTINUE
C
C
      COS1=COS(B*HM1)
      SIN1=-SIN(B*HM1)
      COS2=COS(B*HP1)
      SIN2=SIN(B*HP1)
      COS3=COS(B*HM1T2)
      SIN3=-SIN(B*HM1T2)
      COS4=COS(B*HP1T2)
      SIN4=SIN(B*HP1T2)
      COS5=COS(B*RL1)
C     SIN5=SIN(B*RL1)
      COS6=COS(B*H)
      SIN6=SIN(B*H)
      COS7=COS(B*HPT2)
      SIN7=SIN(B*HPT2)
C
C
      T1A=COS1*(CI(1)+CI(2)-CI(5)-CI(6))
      T1B=SIN1*(-SI(1)+SI(2)+SI(5)-SI(6))
      T2A=COS2*(CI(3)+CI(4)-CI(7)-CI(8))
      T2B=SIN2*(-SI(3)+SI(4)+SI(7)-SI(8))
      T3A=COS3*(-CI(5)-CI(6)+CI(9)+CI(10))
      T3B=SIN3*(SI(5)-SI(6)-SI(9)+SI(10))
      T4A=COS4*(-CI(7)-CI(8)+CI(11)+CI(12))
      T4B=SIN4*(SI(7)-SI(8)-SI(11)+SI(12))
      T5A=2.*COS5*COS6*(-CI(13)-CI(14)+CI(15)+CI(16))
      T5B=2.*COS5*SIN6*(SI(13)-SI(14)-SI(15)+SI(16))
      T6A=2.*COS5*COS7*(CI(15)+CI(16)-CI(17)-CI(18))
      T6B=2.*COS5*SIN7*(-SI(15)+SI(16)+SI(17)-SI(18))
C
      ZR=15*(T1A+T1B+T2A+T2B+T3A+T3B+T4A+T4B+T5A+T5B+T6A+T6B)
```

```
C
C
      T1A=COS1*(-SI(1)-SI(2)+SI(5)+SI(6))
      T1B=SIN1*(-CI(1)+CI(2)+CI(5)-CI(6))
      T2A=COS2*(-SI(3)-SI(4)+SI(7)+SI(8))
      T2B=SIN2*(-CI(3)+CI(4)+CI(7)-CI(8))
      T3A=COS3*(SI(5)+SI(6)-SI(9)-SI(10))
      T3B=SIN3*(CI(5)-CI(6)-CI(9)+CI(10))
      T4A=COS4*(SI(7)+SI(8)-SI(11)-SI(12))
      T4B=SIN4*(CI(7)-CI(8)-CI(11)+CI(12))
      T5A=2.*COS5*COS6*(SI(13)+SI(14)-SI(15)-SI(16))
      T5B=2.*COS5*SIN6*(CI(13)-CI(14)-CI(15)+CI(16))
      T6A=2.*COS5*COS7*(-SI(15)-SI(16)+SI(17)+SI(18))
      T6B=2.*COS5*SIN7*(-CI(15)+CI(16)+CI(17)-CI(18))
C
      ZX=15*(T1A+T1B+T2A+T2B+T3A+T3B+T4A+T4B+T5A+T5B+T6A+T6B)
C
      RETURN
      END
```

```
$PAGESIZE:80
$LINESIZE:132
$MESSAGE:'Compiling KING2'
$SUBTITLE: '    SUBROUTINE KING2'
$PAGE
$NOTRUNCATE
      SUBROUTINE KING2(H,RL1,RL2,ZR,ZX)
C
C     Written by Mike Guler
C     LAST MODIFIED 02-09-89
C
      PARAMETER (PI=3.1415926535898,B=2.*PI)
      REAL U(9),SI(9),CI(9)
      HM1=H-RL1
      HP1=H+RL1
      HP2=H+RL2
      HPT2=H+2.*RL2
      HM12=H-RL1+RL2
      HP12=H+RL1+RL2
      HM1T2=HPT2-RL1
      HP1T2=HPT2+RL1
C
C
      U(1)=2.*B*HM1
      U(2)=2.*B*HP1
      U(3)=2.*B*HM12
      U(4)=2.*B*HP12
      U(5)=2.*B*HM1T2
      U(6)=2.*B*HP1T2
      U(7)=2.*B*H
      U(8)=2.*B*HP2
      U(9)=2.*B*HPT2
C
C
      DO 10 I=1,9
        CALL SICI(U(I),SI(I),CI(I))
   10 CONTINUE
C
C
      COS1=COS(B*HM1)
      SIN1=SIN(B*HM1)
      COS2=COS(B*HP1)
      SIN2=SIN(B*HP1)
      COS3=COS(B*HM1T2)
      SIN3=SIN(B*HM1T2)
      COS4=COS(B*HP1T2)
      SIN4=SIN(B*HP1T2)
      COS5=COS(B*RL1)
C     SIN5=SIN(B*RL1)
      COS6=COS(B*H)
      SIN6=SIN(B*H)
      COS7=COS(B*HPT2)
      SIN7=SIN(B*HPT2)
C
C
      RLOG1=LOG(HM12/HM1)
      RLOG2=LOG(HP12/HP1)
      RLOG3=LOG(HM12/HM1T2)
      RLOG4=LOG(HP12/HP1T2)
      RLOG5=LOG(H/HP2)
      RLOG6=LOG(HPT2/HP2)
C
C
      T1=COS1*(CI(1)-CI(3)+RLOG1)+SIN1*(SI(1)-SI(3))
      T2=COS2*(CI(2)-CI(4)+RLOG2)+SIN2*(SI(2)-SI(4))
      T3=COS3*(CI(5)-CI(3)+RLOG3)+SIN3*(SI(5)-SI(3))
      T4=COS4*(CI(6)-CI(4)+RLOG4)+SIN4*(SI(6)-SI(4))
      T5=2.*(COS5*COS6*(CI(8)-CI(7)+RLOG5)+COS5*SIN6*(SI(8)-SI(7)))
      T6=2.*(COS5*COS7*(CI(8)-CI(9)+RLOG6)+COS5*SIN7*(SI(8)-SI(9)))
```

```
C
      ZR=15.*(T1+T2+T3+T4+T5+T6)
C
C
      T1=SIN1*(CI(1)-CI(3)-RLOG1)+COS1*(SI(3)-SI(1))
      T2=SIN2*(CI(2)-CI(4)-RLOG2)+COS2*(SI(4)-SI(2))
      T3=SIN3*(CI(5)-CI(3)-RLOG3)+COS3*(SI(3)-SI(5))
      T4=SIN4*(CI(6)-CI(4)-RLOG4)+COS4*(SI(4)-SI(6))
      T5=2.*(COS5*SIN6*(CI(8)-CI(7)-RLOG5)+COS5*COS6*(SI(7)-SI(8)))
      T6=2.*(COS5*SIN7*(CI(8)-CI(9)-RLOG6)+COS5*COS7*(SI(9)-SI(8)))
C
      ZX=15.*(T1+T2+T3+T4+T5+T6)
C
      RETURN
      END
```

```
$PAGESIZE:80
$LINESIZE:132
$MESSAGE:'Compiling LDCURNT'
$SUBTITLE: '     SUBROUTINE LDCURNT'
$PAGE
$NOTRUNCATE
      SUBROUTINE LDCURNT(KO,XLOCATE,YLOCATE,NXEL,NXELMX,NYEL,NUMBEAM,
     $  AZBEAM,ELBEAM,PANTILT,DRVCURR)
C
C     LAST MODIFIED 10/18/88.  BY R. WILSON
C     ************** 11/01/88.  BY R. WILSON.
C          1) CHANGED FROM USING "THBEAM" AND "PHIBEAM" (SPHERICAL ANGLES)
C             TO DIRECTLY USING "AZBEAM" AND "ELBEAM".
C     ************** 11/08/88.  BY R. WILSON.
C          1)  ADDED HANDLING OF PANEL TILT.
C     **************  3/09/88.  BY R. WILSON.
C          1)  ADDED HANDLING OF MULTIPLE BEAMS.
C
C     "LDCURNT" COMPUTES THE DRIVING POINT CURRENTS FOR A TWO DIMENSIONAL ARRAY
C     REQUIRED TO POINT THE MAIN BEAM IN THE DIRECTION ("AZBEAM","ELBEAM").
C
C     DESCRIPTION OF PARAMETERS
C
C     "KO" IS THE FREE SPACE WAVENUMBER.  IT IS AN INPUT TO "LDCURNT".
C
C     "XLOCATE" AND "YLOCATE" ARE ONE-DIMENSIONAL ARRAYS DESCRIBING THE POSITION
C     OF EACH OF THE DIPOLES.  FOR EXAMPLE, THE X, Y LOCATION OF THE (2,4)TH
C     ELEMENT IS (XLOCATE(2),YLOCATE(4)).  THE UNITS ARE ASSUMED TO BE COMPATIBLE
C     WITH "KO".  IF "KO" IS IN RADIANS/METER THEN THE UNITS OF "XLOCATE" AND
C     "YLOCATE" ARE METERS.  (THIS IS TAKEN CARE OF AUTOMATICALLY BY "GETGEOM").
C     "XLOCATE" AND "YLOCATE" ARE INPUTS TO "LDCURNT".
C
C     "NXEL" AND "NYEL" ARE THE NUMBER OF DIPOLES ALONG X AND Y RESPECTIVELY.
C     "NXELMX" IS THE MAXIMUM POSSIBLE NUMBER OF DIPOLES ALONG THE "X" AXIS.
C     THIS PARAMETER IS NEEDED IN THE CALL FOR DIMENSIONING THE TWO DIMENSIONAL
C     ARRAYS.
C
C     "NUMBEAM" IS THE NUMBER OF BEAMS DESIRED.  IT IS AN INPUT TO "LDCURNT".
C
C     "AZBEAM" AND "ELBEAM" ARE ARRAYS CONTAINING THE AZIMUTH AND ELEVATION
C     ANGLES, IN DEGREES OF EACH BEAM.  THE "AZBEAM(I)" AND "ELBEAM(I) ARE
C     ASSUMED TO BE IN DEGREES.  "AZBEAM" AND "ELBEAM" ARE INPUTS TO "LDCURNT".
C
C     "PANTILT" IS THE ELEVATION ANGLE OF THE ARRAY PANEL NORMAL IN DEGREES.
C     IT IS AN INPUT TO "LDCURNT".
C
C     "DRVCURR" IS A COMPLEX ARRAY OF DRIVING POINT CURRENTS REQUIRED TO POINT
C     THE MAIN BEAM IN THE DIRECTION SPECIFIED BY "AZBEAM" AND "ELBEAM".
C     "DRVCURR" IS AN OUTPUT FROM "LDCURNT".
C
      COMPLEX DRVCURR(NXELMX,NYEL)
      REAL XLOCATE(NXEL),YLOCATE(NYEL),AZBEAM(NUMBEAM),ELBEAM(NUMBEAM)
      REAL KDIR(3),KO,KYY
C
C     CHECK FOR CORRECT INPUTS.
C
      IF(NXEL .GT. NXELMX)THEN
         CALL BEEP
         WRITE(*,100)
  100    FORMAT(/1X,'***** ERROR - program terminated in LDCURNT.'/
     $      7X,'NXEL(=',I3,') .GT. NXELMX(=',I3,').'/)
         STOP
      ENDIF
C
C     COMPUTE COMPLEX DRIVING POINT CURRENTS TO ACHIEVE THIS BEAM DIRECTION.
C
      DO 5 IBEAM=1,NUMBEAM,1
         CALL AZEL2K(AZBEAM(IBEAM),ELBEAM(IBEAM),PANTILT,KDIR)
         KDIR(1)=KO*KDIR(1)
```

```
      KDIR(2)=K0*KDIR(2)
      IF(IBEAM .EQ. 1)THEN
         DO 2 J=1,NYEL,1
            KYY=KDIR(2)*YLOCATE(J)
            DO 1 I=1,NXEL,1
               DRVCURR(I,J)=
     $            CEXP(CMPLX(0.0,-(KDIR(1)*XLOCATE(I)+KYY)))
 1          CONTINUE
 2       CONTINUE
      ELSE
         DO 4 J=1,NYEL,1
            KYY=KDIR(2)*YLOCATE(J)
            DO 3 I=1,NXEL,1
               DRVCURR(I,J)=DRVCURR(I,J)+
     $            CEXP(CMPLX(0.0,-(KDIR(1)*XLOCATE(I)+KYY)))
 3          CONTINUE
 4       CONTINUE
     ENDIF
 5 CONTINUE
   RETURN
   END
```

```
$PAGESIZE:80
$LINESIZE:132
$MESSAGE:'Compiling MOM'
$SUBTITLE: '    SUBROUTINE MOM'
$PAGE
$NOTRUNCATE
      SUBROUTINE MOM(NWIRES,XOFFSET,YOFFSET,ZOFFSET,EL1,EL2,A,ZOUT)
C
C     Written by Mike Guler
C     LAST MODIFIED 03-31-89
C     ************** 4/11/89. BY R. WILSON.
C          1) ADDED PARAMETER "YOFFSET".
C          2) THIS ENABLES HANDLING OF IMAGE ANTENNA DUE TO A GROUND PLANE.
C     ************** 5/09/89. BY M. GULER.
C          1) CHANGED HANDLING OF SELF TERM TO ONLY CALL "KING1".
C
C     METHOD OF MOMENTS PROGRAM TO CALCULATE CURRENTS ON PARALLEL WIRES OF
C     ARBITRARY LENGTH.  A WIRE RADIUS IS ENTERED AND USED FOR COUPLING BETWEEN
C     SEGMENTS ON THE SAME WIRE, BUT A THIN WIRE IS ASSUMED FOR COUPLING
C     BETWEEN SEGMENTS ON DIFFERENT WIRES.  PIECEWISE SINUSOIDAL MODES ARE USED
C     FOR THE CURRENT EXPANSION FUNCTIONS AND FOR THE WEIGHTING FUNCTIONS.  ALL
C     WIRES ARE ORIENTED PARALLEL TO THE Z-AXIS, AND EACH WIRE HAS A UNIQUE
C     (X,Y) LOCATION.
C
C     SELF AND MUTUAL IMPEDENCE CALCULATIONS USE EQUATIONS FROM
C     H.R. KING, IRE A.&P., JULY 1957, PP 306-313.
C
C     DESCRIPTION OF INPUT VARIABLES (ALL DISTANCES IN WAVELENGTHS)
C
C     NWIRES  -  NUMBER OF WIRES (1 OR 2)
C     XOFFSET -  X-OFFSET OF WIRE 2
C     YOFFSET -  Y-OFFSET OF WIRE 2
C     ZOFFSET -  Z-OFFSET OF WIRE 2  (NOTE: SEGMENTATION IS ALONG THE Z AXIS)
C     EL1     -  LENGTH OF ELEMENT 1
C     EL2     -  LENGTH OF ELEMENT 2
C     A(1)    -  RADIUS OF ELEMENT 1
C     A(2)    -  RADIUS OF ELEMENT 2
C
C  VARIABLE LIST: (ALL DISTANCES MEASURED IN WAVELENGTHS)
C
C   COMPLEX:
C     AMPS(I)     CURRENT AT PULSE I IN AMPS
C     VOLTS(I)    SOURCE VOLTAGE AT PULSE I IN VOLTS
C     YVOLTS(I)   COPY OF SOURCE VOLTAGE FOR USE IN CSOLVE
C     ZM(I,J)     IMPEDANCE MATRIX (BETWEEN PULSES I & J) IN OHMS
C     ZMUT        MUTUAL IMPEDANCE BETWEEN LOAD & SOURCE PULSES IN OHMS
C
C   REAL:
C     A(I)        RADIUS OF WIRE I
C     FACT        FACTOR WHICH ADJUSTS SINUSOIDAL FUNCTION MAGNITUDES
C     H           SEGMENT OFFSET IN Z-DIRECTION (DEFINED IN KING)
C     PI          3.1415926535898
C     R           SEGMENT SEPARATION IN X-DIRECTION (DEFINED IN KING)
C     SEGL(I)     SEGMENT LENGTH FOR WIRE I
C     VMAG        MAGNITUDE OF VOLTAGE SOURCE IN VOLTS
C     VPHS        PHASE ANGLE OF VOLTAGE SOURCE IN DEGREES
C     X(I)        X-LOCATION OF WIRE I
C     Y(I)        Y-LOCATION OF WIRE I
C     Z(I,J)      Z-LOCATION OF THE J-TH PULSE OF WIRE I
C     ZEND(I,1)   MINIMUM Z-LOCATION OF WIRE I
C     ZEND(I,2)   MAXIMUM Z-LOCATION OF WIRE I
C     ZR          REAL PART OF MUTUAL IMPEDANCE IN OHMS (DEFINED IN KING)
C     ZX          IMAGINARY PART OF MUTUAL IMPEDANCE IN OHMS (DEFINED IN KING)
C
C   INTEGER:
C     IMUTP       PULSE NUMBER FOR MUTUAL IMPEDANCE CALCULATION
C     MXDIM       MAXIMUM DIMENSION OF Z-MATRIX
C     MXWIRE      MAXIMUM NUMBER OF WIRES ALLOWED
C     NOS         OFFSET COUNT USED TO FILL Z-MATRIX
```

```
C     NOSI       OFFSET COUNT USED TO FILL Z-MATRIX
C     NOSJ       OFFSET COUNT USED TO FILL Z-MATRIX
C     NP         TOTAL NUMBER OF PULSES
C     NSEG(I)    NUMBER OF SEGMENTS IN WIRE I
C     NSRCP(I)   PULSE NUMBER FOR SOURCE NUMBER I
C     NWIRES     NUMBER OF WIRES
C     P(I)       INDEX ARRAY FOR USE IN CSOLVE
C
C   LOGICAL:
C     PARPIV     PARTIAL PIVOTING SWITCH FOR USE IN CSOLVE
C
C
C  EXTERNALS:
C     KING1, KING2  CALCULATE ZR AND ZX AS DEFINED IN KINGS PAPER
C     SICI    CALCULATES SINE & COSINE INTEGRALS DEFINED IN ABRAMOWITZ & STEGUN
C
C     CSOLVE  COMPLEX MATRIX EQUATION SOLVER WRITTEN BY RICHARD WILSON
C
C
      PARAMETER (MXDIM=18,MXWIRE=2)
      PARAMETER (PI=3.1415926535898)
      COMPLEX ZM(MXDIM,MXDIM),VOLTS(MXDIM),YVOLTS(MXDIM)
      COMPLEX AMPS(MXDIM)
      COMPLEX ZOUT
      REAL SEGL(MXWIRE),Z(MXWIRE,MXDIM),X(MXWIRE),Y(MXWIRE)
      REAL ZEND(MXWIRE,2),A(MXWIRE)
      INTEGER NSEG(MXWIRE),P(MXDIM),NSRCP(MXDIM)
      LOGICAL PARPIV
      PARPIV=.TRUE.
      NP=0
      NSEG(1)=6
      NSEG(2)=6
      X(1)=0.
      X(2)=XOFFSET
      Y(1)=0.
      Y(2)=YOFFSET
      ZEND(1,1)=-EL1/2.
      ZEND(1,2)=EL1/2.
      ZEND(2,1)=ZOFFSET-EL2/2.
      ZEND(2,2)=ZOFFSET+EL2/2.
      NSRCP(1)=NSEG(1)/2
      VMAG=1.
      VPHS=0.
      IF(NWIRES.GT.2.OR.NWIRES.LE.0) THEN
        CALL BEEP
        WRITE(*,100)
  100   FORMAT(/1X,'***** ERROR - program terminated in MOM.')
        WRITE(*,101)NWIRES
  101   FORMAT(7X,'NWIRES(=',I2,') must be 1 or 2.'/)
        STOP
      ENDIF
      IF(NWIRES.EQ.2) IMUTP=NSEG(1)-1+NSEG(2)/2
      IF(EL1.LE.0..OR.EL2.LE.0) THEN
        CALL BEEP
        WRITE(*,100)
        WRITE(*,102)EL1,EL2
  102   FORMAT(7X,'EL1(=',1PE10.3,') and EL2(=',1PE10.3,') must be',
     $    1X,'.GE. 0.0.'/)
        STOP
      ENDIF
      DO 10 I=1,NWIRES
        IF(A(I).LE.0.) THEN
          CALL BEEP
          WRITE(*,100)
          WRITE(*,103)I,A(I)
  103     FORMAT(7X,'Radius of Wire ',I1,'(=',1PE10.3,') must be',
     $      1X,'.GT. 0.0.'/)
          STOP
        ENDIF
```

```
          IF(NSEG(I).LE.1) THEN
            CALL BEEP
            WRITE(*,100)
            WRITE(*,104)I,NSEG(I)
  104       FORMAT(7X,'No. Segments on Wire ',I1,'(=',I3,
     $        ') must be .GT. 1.'/)
            STOP
          ENDIF
          DELZ=ZEND(I,2)-ZEND(I,1)
          SEGL(I)=DELZ/NSEG(I)
          IF(SEGL(I).GT.0.25) THEN
            CALL BEEP
            WRITE(*,*) '*** WARNING ***'
            WRITE(*,*) 'WIRE NUMBER ',I
            WRITE(*,*) 'SEGMENT LENGTH IS GREATER THAN .25 WAVELENGTHS'
          ENDIF
          DO 15 J=1,NSEG(I)-1
            Z(I,J)=ZEND(I,1)+J*SEGL(I)
   15     CONTINUE
          NP=NP+NSEG(I)-1
   10 CONTINUE
      DO 20 I=1,NP
        VOLTS(I)=CMPLX(0.,0.)
        DO 25 J=1,NP
          ZM(I,J)=CMPLX(0.,0.)
   25   CONTINUE
   20 CONTINUE
      VOLTS(NSRCP(1))=CMPLX(VMAG*COS(VPHS),VMAG*SIN(VPHS))
C
C     FILL Z-MATRIX FOR COUPLING BETWEEN SEGMENTS ON SAME WIRE
      NOS=0
      DO 30 I1=1,NWIRES
        FACT=(SIN(2.*PI*SEGL(I1)))**2
        DO 35 I2=1,NSEG(I1)-1
          H=(I2-2)*SEGL(I1)
          CALL KING1(A(I1),H,SEGL(I1),SEGL(I1),ZR,ZX)
          DO 40 I3=1,NSEG(I1)-I2
            IF(I2.EQ.1) THEN
              K1=I3+NOS
              ZM(K1,K1)=CMPLX(ZR,ZX)/FACT
            ELSE
              K1=I3+NOS+I2-1
              K2=I3+NOS
              ZM(K1,K2)=CMPLX(ZR,ZX)/FACT
              ZM(K2,K1)=ZM(K1,K2)
            ENDIF
   40     CONTINUE
   35   CONTINUE
        NOS=NOS+NSEG(I1)-1
   30 CONTINUE
C
C     FILL Z-MATRIX FOR COUPLING BETWEEN SEGMENTS ON DIFFERENT WIRES
      DO 45 I1=1,NWIRES
        NOSI=NOSSET(I1,NSEG)
        DO 50 J1=I1+1,NWIRES
          FACT=SIN(2.*PI*SEGL(I1))*SIN(2.*PI*SEGL(J1))
          R=SQRT((X(I1)-X(J1))**2+(Y(I1)-Y(J1))**2)
          NOSJ=NOSSET(J1,NSEG)
          DO 55 J2=1,NSEG(J1)-1
            I2=1
            K2=J2
            H=ABS(Z(J1,J2)-Z(I1,I2))-SEGL(J1)
            IF(R.GT.1.E-8) THEN
              CALL KING1(R,H,SEGL(I1),SEGL(J1),ZR,ZX)
            ELSE
              CALL KING2(H,SEGL(I1),SEGL(J1),ZR,ZX)
            ENDIF
  500       ZM(K2+NOSJ,I2+NOSI)=CMPLX(ZR,ZX)/FACT
            ZM(I2+NOSI,K2+NOSJ)=ZM(K2+NOSJ,I2+NOSI)
```

```
            I2=I2+1
            IF(I2.GE.NSEG(I1)) GOTO 55
            K2=K2+1
            IF(K2.GE.NSEG(J1)) GOTO 55
            GOTO 500
   55     CONTINUE
          IF(NSEG(I1).NE.2) THEN
            DO 60 I2=2,NSEG(I1)-1
              J2=1
              K2=I2
              H=ABS(Z(J1,J2)-Z(I1,I2))-SEGL(J1)
              IF(R.GT.1.E-8) THEN
                CALL KING1(R,H,SEGL(I1),SEGL(J1),ZR,ZX)
              ELSE
                CALL KING2(H,SEGL(I1),SEGL(J1),ZR,ZX)
              ENDIF
  510         ZM(J2+NOSJ,K2+NOSI)=CMPLX(ZR,ZX)/FACT
              ZM(K2+NOSI,J2+NOSJ)=ZM(J2+NOSJ,K2+NOSI)
              J2=J2+1
              IF(J2.GE.NSEG(J1)) GOTO 60
              K2=K2+1
              IF(K2.GE.NSEG(I1)) GOTO 60
              GOTO 510
   60       CONTINUE
          ENDIF
   50   CONTINUE
   45 CONTINUE
C
C
C  ADJUST Z-MATRIX FOR MUTUAL CALCULATION
      IF(NWIRES.EQ.2) THEN
        DO 62 I=1,NP
          ZM(I,IMUTP)=CMPLX(0.,0.)
   62   CONTINUE
        ZM(IMUTP,IMUTP)=CMPLX(-1.,0.)
      ENDIF
C
C
      DO 64 I=1,NP
        YVOLTS(I)=VOLTS(I)
   64 CONTINUE
      CALL CSOLVE(ZM,MXDIM,NP,YVOLTS,PARPIV,P,AMPS)
      IF(NWIRES.EQ.1) THEN
        ZOUT=VOLTS(NSRCP(1))/AMPS(NSRCP(1))
      ELSE
        ZOUT=AMPS(IMUTP)/AMPS(NSRCP(1))
      ENDIF
      END
```

```
$PAGESIZE:80
$LINESIZE:132
$MESSAGE:'Compiling NOSSET'
$SUBTITLE: '    FUNCTION NOSSET'
$PAGE
$NOTRUNCATE
      FUNCTION NOSSET(IWIRE,NSEG)
C
C     Written by Mike Guler
C     LAST MODIFIED 02-09-89
C
      INTEGER NSEG(*)
      NOSSET=0
      IF(IWIRE.EQ.1) RETURN
      DO 10 I=1,IWIRE-1
        NOSSET=NOSSET+NSEG(I)-1
   10 CONTINUE
      RETURN
      END
```

```
$PAGESIZE:80
$LINESIZE:132
$MESSAGE:'Compiling SHOWIN'
$SUBTITLE: '    SUBROUTINE SHOWIN'
$PAGE
$NOTRUNCATE
      SUBROUTINE SHOWIN(UNITS,FREQGHZ,NUMBEAM,AZBEAM,ELBEAM,PANTILT,
     $  DIPTILT,LFILNET,GNDPLN,XSIZE,YSIZE,ERSUB,THSUB,INPUTOK)
C
C   LAST MODIFIED  9/13/89.  BY R. WILSON.
C
C   "SHOWIN" displays the inputs to be used in the array design for approval
C   by the user.
C
      REAL AZBEAM(NUMBEAM),ELBEAM(NUMBEAM)
      REAL GNDPLN(3)
      CHARACTER CNUM*80,EFILNET*6,UNITS*2,LFILNET*1
      LOGICAL YESNOTF,INPUTOK
C
C   Write a header.
C
      CALL NUMCHAR(FREQGHZ,-14,0,CNUM,NCHAR)
      WRITE(*,100)CNUM(1:NCHAR)
  100 FORMAT(/1X,'The parameters which will be used in the design of',
     $  1X,'the array are as follows:'//5X,'Source Frequency',
     $  1X,22('.'),1X,A,' GHz')
C
C   Write beam positions.
C
      DO 1 I=1,NUMBEAM,1
         CALL NUMCHAR(FLOAT(I),0,0,CNUM,NCHAR)
         CALL NUMCHAR(AZBEAM(I),-14,NCHAR,CNUM,NCHAR1)
         CALL NUMCHAR(ELBEAM(I),-14,NCHAR1,CNUM,NCHAR2)
         WRITE(*,101)CNUM(1:NCHAR),CNUM(NCHAR+1:NCHAR1),
     $     CNUM(NCHAR1+1:NCHAR2)
  101    FORMAT(5X,'Position of Beam #',A,' (Az El)',1X,11('.'),
     $     1X,A,2X,A,' degrees')
    1 CONTINUE
C
C   Write panel tilt and dipole tilt.
C
      CALL NUMCHAR(PANTILT,-14,0,CNUM,NCHAR1)
      CALL NUMCHAR(DIPTILT,-14,NCHAR1,CNUM,NCHAR2)
      WRITE(*,102)CNUM(1:NCHAR1),CNUM(NCHAR1+1:NCHAR2)
  102 FORMAT(5X,'Panel Tilt ',28('.'),1X,A,' degrees'/
     $  5X,'Dipole Tilt ',27('.'),1X,A,' degrees')
C
C   Write location of filter network.
C
      IF(LFILNET .EQ. 'T')THEN
         EFILNET='Top'
      ELSEIF(LFILNET .EQ. 'B')THEN
         EFILNET='Bottom'
      ELSEIF(LFILNET .EQ. 'L')THEN
         EFILNET='Left'
      ELSEIF(LFILNET .EQ. 'R')THEN
         EFILNET='Right'
      ENDIF
      WRITE(*,103)EFILNET
  103 FORMAT(5X,'Location of the Filter Network ',8('.'),1X,A)
C
C   Write Ground plane info.
C
      IF(GNDPLN(1) .GT. 0.0)THEN
         CALL NUMCHAR(GNDPLN(3),-14,0,CNUM,NCHAR)
         WRITE(*,104)CNUM(1:NCHAR)
  104    FORMAT(5X,'Gnd Plane Spacer Dielectric Constant ..',
     $     1X,A)
      ELSE
```

```
            WRITE(*,105)
  105       FORMAT(5X,'Gnd Plane Spacer Dielectric Constant .. no',
     $         1X,'ground plane')
        ENDIF
C
C   Write panel dimensions.
C
        CALL NUMCHAR(XSIZE,-14,0,CNUM,NCHAR1)
        CALL NUMCHAR(YSIZE,-14,NCHAR1,CNUM,NCHAR2)
        WRITE(*,106)CNUM(1:NCHAR1),UNITS,CNUM(NCHAR1+1:NCHAR2),UNITS
  106   FORMAT(5X,'Panel horizontal dimension ',12('.'),1X,A,1X,A2,'.'/
     $   5X,'Panel vertical dimension ',14('.'),1X,A,1X,A2,'.')
C
C   Write substrate dielectric constant and substrate thickness.
C
        CALL NUMCHAR(ERSUB,-14,0,CNUM,NCHAR1)
        CALL NUMCHAR(THSUB,-14,NCHAR1,CNUM,NCHAR2)
        WRITE(*,107)CNUM(1:NCHAR1),CNUM(NCHAR1+1:NCHAR2),UNITS
  107   FORMAT(5X,'Substrate Dielectric Constant ',9('.'),1X,A/
     $   5X,'Substrate thickness ',19('.'),1X,A,1X,A2,'.'/)
C
C   Ask user if these are okay.
C
        INPUTOK=YESNOTF('Are these parameters ok')
        RETURN
        END
```

```
$PAGESIZE:80
$LINESIZE:132
$MESSAGE:'Compiling SICI'
$SUBTITLE: '    SUBROUTINE SICI'
$PAGE
$NOTRUNCATE
      SUBROUTINE SICI(X,SI,CI)
C
C     Written by Mike Guler
C     Last Modified 02-15-89
C
      PARAMETER(A1=38.027264,A2=265.187033,A3=335.677320,A4=38.102495)
      PARAMETER(B1=40.021433,B2=322.624911,B3=570.236280,B4=157.105423)
      PARAMETER(C1=42.242855,C2=302.757865,C3=352.018498,C4=21.821899)
      PARAMETER(D1=48.196927,D2=482.485984,D3=1114.978885,D4=449.690326)
      PARAMETER (PI=3.1415926535898,PIOVER2=PI/2.0,EC=.57721566490153)
      REAL X,SII,CII,SI,CI
      IF(X .LE. 0.0) THEN
        CALL BEEP
        WRITE(*,100)X
  100   FORMAT(/1X,'***** ERROR - Program terminated in SICI.'/
     $    7X,'X(=',1PE10.3,') must be .GT. 0.'/)
        STOP
      ENDIF
      IF(X.LT.1.) THEN
        N=NINT(2.*X+2.4)
        SI=X
        CI=0
        DO 10 I=1,N
          SII=(-1)**I*X**(2*I+1)/(2*I+1)
          CII=(-1)**I*X**(2*I)/(2*I)
          DO 20 J1=2,2*I+1
            SII=SII/J1
   20     CONTINUE
          DO 30 J2=2,2*I
            CII=CII/J2
   30     CONTINUE
          SI=SI+SII
          CI=CI+CII
   10   CONTINUE
        CI=CI+LOG(X)+EC
      ELSE
        X2=X*X
        RNUM=A4+X2*(A3+X2*(A2+X2*(A1+X2)))
        DNUM=B4+X2*(B3+X2*(B2+X2*(B1+X2)))
        F=RNUM/DNUM/X
        RNUM=C4+X2*(C3+X2*(C2+X2*(C1+X2)))
        DNUM=D4+X2*(D3+X2*(D2+X2*(D1+X2)))
        G=RNUM/DNUM/X2
        SI=PIOVER2-F*COS(X)-G*SIN(X)
        CI=F*SIN(X)-G*COS(X)
      ENDIF
      RETURN
      END
```

```
$PAGESIZE:80
$LINESIZE:132
$MESSAGE:'Compiling SQROOT'
$SUBTITLE: '    SUBROUTINE SQROOT'
$PAGE
$NOTRUNCATE
      SUBROUTINE SQROOT(D,H,U1,U2)
C
C     Written by Mike Guler
C     LAST MODIFIED 02-07-89
C
      PARAMETER (PI=3.1415926535898,B=2.*PI)
      SQR=SQRT(D*D+H*H)
      IF(H.GT.0) THEN
        U1=B*(SQR+H)
        U2=B*D*D/(SQR+H)
      ELSEIF(H.LT.0) THEN
        U1=B*D*D/(SQR-H)
        U2=B*(SQR-H)
      ELSE
        U1=B*D
        U2=U1
      ENDIF
      RETURN
      END
```

```
$PAGESIZE:80
$LINESIZE:132
$MESSAGE:'Compiling SUMMARY'
$SUBTITLE: '    SUBROUTINE SUMMARY'
$PAGE
$NOTRUNCATE
      SUBROUTINE SUMMARY(DATSTMP,UNITS,FREQGHZ,LAMBDA,NUMBEAM,AZBEAM,
     $ ELBEAM,PANTILT,XSIZE,YSIZE,NXEL,NXELMX,NYEL,DX,DY,DIPLNG,DIPWID,
     $ DIPTILT,LFILNET,ERSUB,THSUB,GNDPLN,XLOCATE,YLOCATE,DRVCURR,
     $ ZDRVIMP, GAIN)
C
C   LAST MODIFIED  9/13/89.  BY R. WILSON.
C
C   "SUMMARY" writes a summary of the array design to a file.
C
      COMPLEX ZDRVIMP(NXELMX,NYEL),DRVCURR(NXELMX,NYEL)
      COMPLEX REC2POL
      REAL XLOCATE(NXEL),YLOCATE(NYEL)
      REAL AZBEAM(NUMBEAM),ELBEAM(NUMBEAM),GAIN(NUMBEAM),DIPTILT(3)
      REAL GNDPLN(3),LAMBDA
      CHARACTER DATSTMP*20,UNITS*2,LFILNET*1
C
C   Write the header to the file.
C
      WRITE(1,*)'DIPOLE ARRAY DESIGN             ',DATSTMP
      WRITE(1,*)'                   *****'
      WRITE(1,*)'The operating frequency is ',FREQGHZ,' GHz',
     $ '    (Lambda=',LAMBDA,' ',UNITS,'.)'
      IF(NUMBEAM .EQ. 1)THEN
         WRITE(1,*)'The beam is pointed at ',AZBEAM(1),' deg. AZIMUTH ',
     $      ELBEAM(1),' deg. ELEVATION'
      ELSE
         WRITE(1,*)'The beams are pointed at ',AZBEAM(1),' deg. ',
     $      'AZIMUTH ',ELBEAM(1),' deg. ELEVATION'
         DO 1 IBEAM=2,NUMBEAM,1
         WRITE(1,*)'                         ',AZBEAM(IBEAM),
     $      ' deg. AZIMUTH ',ELBEAM(IBEAM),' deg. ELEVATION'
    1    CONTINUE
      ENDIF
      WRITE(1,*)'                   *****'
      WRITE(1,*)'The array panel is tilted ',PANTILT,' degrees in ',
     $ 'ELEVATION'
      WRITE(1,*)'The array is ',XSIZE,' ',UNITS,'. by ',
     $ YSIZE,' ',UNITS,'.'
      WRITE(1,*)'There are ',NXEL,' elements in the HORIZONTAL',
     $ ' direction'
      WRITE(1,*)'          ',NYEL,' elements in the VERTICAL direction'
      WRITE(1,*)'The dipole elements are spaced ',DX,' ',UNITS,
     $ '. apart HORIZONTALLY'
      WRITE(1,*)'                               ',DY,' ',UNITS,
     $ '. apart VERTICALLY'
      WRITE(1,*)'Each dipole is ',DIPLNG,' ',UNITS,'. long and ',DIPWID,
     $ ' ',UNITS,'. wide'
      WRITE(1,*)'Each dipole is tilted ',DIPTILT(3),' degrees from',
     $ ' VERTICAL'
      WRITE(1,*)'                   *****'
      IF(LFILNET .EQ. 'B')THEN
         WRITE(1,*)'The filter network is located along the ',
     $      'BOTTOM edge of the substrate'
      ELSEIF(LFILNET .EQ. 'T')THEN
         WRITE(1,*)'The filter network is located along the ',
     $      'TOP edge of the substrate'
      ELSEIF(LFILNET .EQ. 'R')THEN
         WRITE(1,*)'The filter network is located along the ',
     $      'RIGHT edge of the substrate'
      ELSEIF(LFILNET .EQ. 'L')THEN
         WRITE(1,*)'The filter network is located along the ',
     $      'LEFT edge of the substrate'
      ENDIF
```

```
      WRITE(1,*)'The substrate dielectric constant is ',ERSUB
      WRITE(1,*)'The substrate is ',THSUB,' ',UNITS,'. thick'
      IF(GNDPLN(2) .GT. 0.0)THEN
         WRITE(1,*)'                  *****'
         WRITE(1,*)'A ground plane is to be used ',GNDPLN(2),' ',UNITS,
     $      '. from the array panel'
         WRITE(1,*)'The dielectric constant of the spacer material',
     $      ' is ',GNDPLN(3)
      ENDIF
      WRITE(1,*)'                  *****'
C
C   Write the dipole locations, driving point voltages, currrents and
C   impedances to the file.
C
      WRITE(1,101)
  101 FORMAT(/7X,'Array',5X,'|',8X,'Location',7X,'|',
     $  16X,'Driving Point'/3X,'Column',3X,'Row',2X,'|','Horizontal',
     $  3X,'Vertical',2X,'|',7X,'Current',16X,'Impedance')
      DO 3 J=1,NYEL,1
         Y=YLOCATE(J)
         DO 2 I=1,NXEL,1
            WRITE(1,102)I,J,XLOCATE(I),Y,
     $         REC2POL(DRVCURR(I,J),.FALSE.),
     $         REC2POL(ZDRVIMP(I,J),.FALSE.)
  102       FORMAT(4X,I5,' |',I5,2(' |',G10.3),
     $         2(1X,'|',G10.3,'/_',G10.3))
    2    CONTINUE
    3 CONTINUE
      WRITE(1,*)'                  *****'
C
C   Write the gain values at the main beam locations to the file.
C
      WRITE(1,103)
  103 FORMAT(1X,'Azimuth',2X,'Elevation',2X,'Gain dBi')
      DO 4 I=1,NUMBEAM,1
         WRITE(1,104)AZBEAM(I),ELBEAM(I),GAIN(I)
  104    FORMAT(1X,F8.3,2X,F8.3,2X,F8.3)
    4 CONTINUE
      WRITE(1,*)'                  *****'
C
      RETURN
      END
```

```
$PAGESIZE:80
$LINESIZE:132
$MESSAGE:'Compiling UNIT2IN'
$SUBTITLE: '    FUNCTION UNIT2IN'
$PAGE
$NOTRUNCATE
      FUNCTION UNIT2IN(X,UNITS)
C
C   LAST MODIFIED 5/08/89.
C
      CHARACTER UNITS*2
C
C   "UNIT2IN" CONVERTS "X" IN UNITS OF "UNITS" TO X IN UNITS OF INCHES.
C
      IF(UNITS .EQ. 'ft')THEN
         UNIT2IN=X*12.0
      ELSEIF(UNITS .EQ. 'in')THEN
         UNIT2IN=X
      ELSEIF(UNITS .EQ. 'm')THEN
         UNIT2IN=X/0.0254
      ELSEIF(UNITS .EQ. 'cm')THEN
         UNIT2IN=X/2.54
      ENDIF
      RETURN
      END
```

```
$PAGESIZE:80
$LINESIZE:132
$MESSAGE:'Compiling AXISXY'
$SUBTITLE: '    SUBROUTINE AXISXY'
$PAGE
$NOTRUNCATE
      SUBROUTINE AXISXY(NOPT,XSIZE,YSIZE,BOXDIR,XMAX,XMIN,YMAX,YMIN,
     $  DELTAX,DELTAY,AXISLBL,XSCALE,YSCALE)
C
C   LAST MODIFIED  9/21/87.  BY R. WILSON.
C        1) DETERMINE FONT USED FOR AXIS NUMBERS WITH CALL TO "SYMESC"
C   *************  4/20/88.  BY R. WILSON.
C        1) NOW DRAW BOX BY CALL TO "DRWBOX".
C   ************* 12/05/88.  BY R. WILSON.
C        1) REMOVED CALL TO "DRWBOX" AND DRAW THE AXES WITHIN "LINAXIS" AND
C           "LOGAXIS".
C        2) REMOVED "CHARTYP" AS ARGUMENT TO "LINAXIS" AND "LOGAXIS".
C   *************  1/10/89.  BY R. WILSON.
C        1) MOVED ADJUSTMENT FOR "YMAX"="YMIN" HERE.
C   *************  5/05/89.  BY R. WILSON.
C        1) ADDED CAPABILITY OF ROTATING THE GRAPH "BOX".
C        2) THE ROTATION IS INDICATED BY THE ARRAY "BOXDIR" WHICH CONTAINS THE
C           DIRECTION COSINES OF THE BOX X-AXIS WITH THE PLOTTER AXES.
C
      REAL XORIGIN(2),BOXDIR(3)
      CHARACTER AXISLBL(2)*(*),IAXIS*8
      COMMON/LINAXC/MAXDIG
C
C   "NOPT" INDICATES HOW THE AXES ARE TO BE LABELED AS FOLLOWS
C                       X-LOW   X-UP   Y-LEFT   Y-RIGHT
C      NOPT=0            PARL    NO     PERP      NO
C      NOPT=1            PARL   TICS    PERP     TICS
C      NOPT=2            PARL   PARL    PERP     PERP
C      NOPT=3            PARL    NO     PARL      NO
C      NOPT=4            PARL   TICS    PARL     TICS
C      NOPT=5            PARL   PARL    PARL     PARL
C
C   WHERE X-LOW REFERS TO THE LOWER X-AXIS, X-UP REFERS TO THE UPPER X-AXIS,
C   Y-LEFT REFERS TO THE LEFT Y-AXIS AND Y-RIGHT REFERS TO THE RIGHT Y-AXIS.
C   "NO" INDICATES NO LABELING ON THAT AXIS, "TICS" INDICATES ONLY TIC MARKS
C   ON THAT AXIS, "PARL" INDICATES TICS MARKS WITH NUMBERS PARALLEL TO THE
C   AXIS AND "PERP" INDICATES TIC MARKS WITH NUMBERS PERPENDICULAR TO THE
C   AXIS.
C
C   FOR DEFINITIONS OF THE OTHER PARAMETERS SEE GRAPH.
C
C   CALCULATE PARAMETERS FOR SYMBOL.
C
      TICSIZE=0.02*SQRT(XSIZE*XSIZE+YSIZE*YSIZE)
      HEIGHT=TICSIZE
      EHT=0.8*HEIGHT
C
C   CHECK FOR "YMAX"="YMIN" AND ADJUST IF NECESSARY.
C
      IF(YMAX .EQ. YMIN)THEN
         DELTAY=1.0/(10.0**(NUMDECR(YMAX,MAXDIG-1)+1))
         YMAX=YMAX+DELTAY
         YMIN=YMIN-DELTAY
      ENDIF
C
C   ANNOTATE LEFT Y-AXIS.
C
      XORIGIN(1)=-YSIZE*BOXDIR(2)
      XORIGIN(2)=YSIZE*BOXDIR(1)
      ANGLE=BOXDIR(3)-90.0
      IF(NOPT.GE.0 .AND. NOPT.LE.2)THEN
         IAXIS='PERP-1-1'
      ELSEIF(NOPT.GE.3 .AND. NOPT.LE.5)THEN
         IAXIS='PARL-1-1'
```

```
      ENDIF
      IF(DELTAY .GE. 0.0)THEN
         CALL LINAXIS(YMIN,YMAX,DELTAY,HEIGHT,TICSIZE,ANGLE,YSIZE,
     $     XORIGIN,IAXIS,YSCALE,YLPRIME)
      ELSEIF(DELTAY .LT. 0.0)THEN
         CALL LOGAXIS(YMIN,YMAX,EHT,HEIGHT,TICSIZE,ANGLE,YSIZE,
     $     XORIGIN,IAXIS,YSCALE,YLPRIME)
      ENDIF
      CALL TITLE(AXISLBL(2),ANGLE,-1,YLPRIME,HEIGHT,YSIZE,XORIGIN)
C
C    ANNOTATE LOWER X-AXIS.
C
      XORIGIN(1)=0.0
      XORIGIN(2)=0.0
      ANGLE=BOXDIR(3)
      IF(DELTAX .GE. 0.0)THEN
         CALL LINAXIS(XMAX,XMIN,DELTAX,HEIGHT,TICSIZE,ANGLE,XSIZE,
     $     XORIGIN,'PARL+1-1',XSCALE,YLPRIME)
      ELSEIF(DELTAX .LT. 0.0)THEN
         CALL LOGAXIS(XMAX,XMIN,EHT,HEIGHT,TICSIZE,ANGLE,XSIZE,
     $     XORIGIN,'PARL+1-1',XSCALE,YLPRIME)
      ENDIF
      CALL TITLE(AXISLBL(1),ANGLE,1,YLPRIME,HEIGHT,XSIZE,XORIGIN)
      IF(NOPT.NE.0 .AND. NOPT.NE.3)THEN
C
C       ADD RIGHT Y-AXIS.
C
         XORIGIN(1)=XSIZE*BOXDIR(1)
         XORIGIN(2)=XSIZE*BOXDIR(2)
         ANGLE=BOXDIR(3)+90.0
         IF(NOPT.EQ.1 .OR. NOPT.EQ.4)THEN
            IAXIS='NOLABEL'
         ELSEIF(NOPT .EQ. 2)THEN
            IAXIS='PERP+1-1'
         ELSEIF(NOPT .EQ. 5)THEN
            IAXIS='PARL-1-1'
         ENDIF
         IF(DELTAY .GE. 0.0)THEN
            CALL LINAXIS(YMAX,YMIN,DELTAY,HEIGHT,TICSIZE,ANGLE,YSIZE,
     $        XORIGIN,IAXIS,YSCALE,YLPRIME)
         ELSEIF(DELTAY .LT. 0.0)THEN
            CALL LOGAXIS(YMAX,YMIN,EHT,HEIGHT,TICSIZE,ANGLE,YSIZE,
     $        XORIGIN,IAXIS,YSCALE,YLPRIME)
         ENDIF
C
C       ADD UPPER X-AXIS.
C
         XORIGIN(1)=XSIZE*BOXDIR(1)-YSIZE*BOXDIR(2)
         XORIGIN(2)=XSIZE*BOXDIR(2)+YSIZE*BOXDIR(1)
         ANGLE=BOXDIR(3)+180.0
         IF(NOPT.EQ.1 .OR.NOPT.EQ.4)THEN
            IAXIS='NOLABEL'
         ELSE
            IAXIS='PARL-1-1'
         ENDIF
         IF(DELTAX .GE. 0.0)THEN
            CALL LINAXIS(XMIN,XMAX,DELTAX,HEIGHT,TICSIZE,ANGLE,XSIZE,
     $        XORIGIN,IAXIS,XSCALE,YLPRIME)
         ELSEIF(DELTAX .LT. 0.0)THEN
            CALL LOGAXIS(XMIN,XMAX,EHT,HEIGHT,TICSIZE,ANGLE,XSIZE,
     $        XORIGIN,IAXIS,XSCALE,YLPRIME)
         ENDIF
      ENDIF
      RETURN
      END
```

```
$PAGESIZE:80
$LINESIZE:132
$MESSAGE:'Compiling BEEP'
$SUBTITLE: '    SUBROUTINE BEEP'
$PAGE
$NOTRUNCATE
      SUBROUTINE BEEP
C
C   LAST MODIFIED 12/07/88.  BY R. WILSON.
C
C   "BEEP" BEEPS THE SPEAKER BY SENDING A CONTROL G TO THE SCREEN.
C
      WRITE(*,'(1X,A1)')CHAR(7)
      RETURN
      END
```

```
$PAGESIZE:80
$LINESIZE:132
$MESSAGE:'Compiling CLOSPLT'
$SUBTITLE: '    SUBROUTINE CLOSPLT'
$PAGE
$NOTRUNCATE
      SUBROUTINE CLOSPLT
C
C   LAST MODIFIED  5/17/89.  BY R. WILSON.
C   *************  9/10/89.  BY R. WILSON.
C        1) Added FACTR to the PLOT88C common block.  This allows NEWPAGE
C           to reset the user specified scale factor for each page of plots.
C        2) Added special handling for the case of graphics output to the
C           console (for 93 <= MODEL <=97 and MODEL=99).  This allows
C           displaying a message for the final page of plots.  The message
C           for the other pages is handled by NEWPAGE.
C        3) Changed the criterion for output to the laser printer to include
C           all the laser printer model numbers.
C   *************  9/11/89. BY R. WILSON.
C        1) XPAGEORG and YPAGEORG are passed in a common block from NEWPAGE
C           for use by CLOSPLT in displaying a message if the console is the
C           output device.
C        2) Changed from IF..THEN to SELECT CASE structure to test for MODEL
C           number.
C
      COMMON/PLOT88C/IOPORT,MODEL,FACTR
      COMMON/NEWPAGC/XPAGEORG,YPAGEORG
C
C   "CLOSPLT" writes a message to the screen if plot is to go to the
C   laser printer or the plotter then closes the PLOT88 file which causes
C   the hardcopy to be displayed.
C
      SELECT CASE (MODEL)
         CASE (60:65)
            WRITE(*,100)CHAR(7)
  100       FORMAT('+',A,'Please wait - sending the plots to the Laser',
     $         1X,'Printer...',20(' '))
            CALL PLOT(0.0,0.0,999)
            WRITE(*,102)CHAR(7)
         CASE (20)
            WRITE(*,101)CHAR(7)
  101       FORMAT('+',A,'Ready to send plots to HP 7470A.  LOAD',
     $         1X,'paper, STRIKE any key when ready...')
            CALL WAITKY(IASCII,ISCANCD)
            WRITE(*,102)' '
            CALL PLOT(0.0,0.0,999)
         CASE(93:97,99)
            IF(MODEL .EQ. 96)CALL COLOR(4,IERR)
            CALL FACTOR(1.0)
            CALL SYMBOL(2.5-XPAGEORG,5.0-YPAGEORG,0.12,
     $         'Strike Any Key To Continue',0.0,26)
            WRITE(*,102)CHAR(7)
            CALL PLOT(0.0,0.0,999)
         CASE DEFAULT
            WRITE(*,102)CHAR(7)
            CALL PLOT(0.0,0.0,999)
      END SELECT
  102 FORMAT('+',A,77(' '))
C
      RETURN
      END
```

```
$PAGESIZE:80
$LINESIZE:132
$MESSAGE:'Compiling COSD'
$SUBTITLE: '    FUNCTION COSD'
$PAGE
$NOTRUNCATE
      FUNCTION COSD(ANGLE)
C
C   LAST MODIFIED 12/01/88.  BY. R. WILSON.
C
C   "COSD" COMPUTES THE COSINE OF THE ANGLE "ANGLE" WHERE "ANGLE" IS IN
C   DEGREES.
C
      PARAMETER(PI=3.1415926535898,DEG2RAD=PI/180.0)
      COSD=COS(ANGLE*DEG2RAD)
      RETURN
      END
```

```
$MESSAGE:'Compiling DATE'
$SUBTITLE: '   Function DATE'
$PAGE
$NOTRUNCATE
      FUNCTION DATE()
C
C   LAST MODIFIED 8/28/88.  BY R. WILSON
C
C   "DATE" RETURNS THE CURRENT DATE (AS REPORTED BY THE SYSTEM CLOCK).  MS
C   FORTRAN HAS A LIBRARY ROUTINE CALLED "GETDAT" WHICH IS CALLED BY "DATE".
C   THE RETURNED INTEGER*2 VALUES ARE PACKED INTO A CHARACTER VARIABLE "DATE".
C
      INTEGER*2 IYR,IMON,IDAY
      CHARACTER DATE*10
C
C   GET THE CURRENT DATE FROM SYSTEM CLOCK.
C
      CALL GETDAT(IYR,IMON,IDAY)
C
C   PACK DATE INTO CHARACTER VARIABLE "DATE".
C
      WRITE(DATE,100)IMON,IDAY,IYR
  100 FORMAT(I2,'/',I2,'/',I4)
      IF(DATE(4:4) .EQ. ' ')THEN
          DATE(4:4)='0'
      ENDIF
      RETURN
      END
```

```
$PAGESIZE:80
$LINESIZE:132
$MESSAGE:'Compiling DBAMP'
$SUBTITLE: '    SUBROUTINE DBAMP'
$PAGE
$NOTRUNCATE
      SUBROUTINE DBAMP(AMP,NX,NY,NXMAX,AMAX,AMINDB,LOGTODB)
C
C   LAST MODIFIED  3/18/86.  BY R. WILSON.
C   *************  6/17/88.  BY R. WILSON.
C        1) ADDED CHECK FOR CORRECT "NX" SIZE COMPARED TO "NXMAX".
C   *************  6/29/89.  BY R. WILSON.
C        1) ADDED CODE TO HANDLE "AMAX" BEING SMALL (PARTICULARLY "AMAX"=0.0).
C        2) A NEW VARIABLE, "ANORM" IS SET TO "AMAX" UNLESS "AMAX" IS .LT.
C           "ZERO" IN WHICH CASE "ANORM" IS SET TO "ZERO".  "ZERO" IS DEFINED IN
C           A PARAMETER STATEMENT BELOW.
C        3) UPDATED THE ERROR MESSAGE FORMAT TO BE CONSISTENT WITH THOSE IN
C           OTHER LIBRARY ROUTINES.
C
C   "DBAMP" CONVERTS THE REAL ARRAY "AMP" TO DB.  THE VALUES IN "AMP" ARE
C   NORMALIZED TO "AMAX" BEFORE CONVERSION.  IF THE RESULTING DB VALUE WILL BE
C   LESS THAN "AMINDB" DB THEN IT IS SET EQUAL TO "AMINDB".  "LOGTODB" IS
C   USUALLY SET TO EITHER 10.0 OR 20.0.  THIS ALLOWS THE CONVERSION OF EITHER
C   POWER RATIOS OR VOLTAGE (FIELD) RATIOS, RESPECTIVELY, TO DB.
C
C   THE ARRAY "AMP" IN THE CALLING PROGRAM MAY BE EITHER 1 OR 2 DIMENSIONAL.
C   "NX" BY "NY" VALUES IN THE ARRAY ARE CONVERTED.  IF "AMP" IS 1 DIMENSIONAL
C   THEN "NY" SHOULD BE PASSED AS A 1 (ONE).  "NXMAX" IS USED BY "DBAMP" AS
C   THE ROW DIMENSION OF "AMP".  IT SHOULD BE SET TO THE VALUE OF THE ROW
C   DIMENSION OF THE CORRESPONDING ARRAY IN THE CALLING PROGRAM, ESPECIALLY IF
C   "AMP" IS TO BE 2 DIMENSIONAL.  A CHECK IS MADE TO INSURE THAT "NX" .LE.
C   "NXMAX" BY "DBAMP" AND THE PROGRAM IS TERMINATED WITH AN ERROR MESSAGE IF
C   THIS IS NOT THE CASE.
C
      PARAMETER (ZERO=1.0E-12)
      REAL AMP(NXMAX,*),LOGTODB
C
C   CHECK FOR PROPER INPUTS.
C
      IF(NX .GT. NXMAX)THEN
         WRITE(*,100)NX,NXMAX
  100    FORMAT(1X,'***** ERROR- Program terminated in DBAMP.'/
     $      7X,'NX(=',I5,') .GT. NXMAX(=',I5,').')
         STOP
      ENDIF
C
C   CONVERT AMP TO DB.
C
      AMIN10=10.0**(AMINDB/LOGTODB)
      ANORM=AMAX
      IF(ANORM .LT. ZERO)ANORM=ZERO
      DO 1 J=1,NY,1
         DO 1 I=1,NX,1
            AIJ=ABS(AMP(I,J)/ANORM)
            IF(AIJ .LE. AMIN10)THEN
               AMP(I,J)=AMINDB
            ELSE
               AMP(I,J)=LOGTODB*ALOG10(AIJ)
            ENDIF
    1 CONTINUE
      RETURN
      END
```

```
$PAGESIZE:80
$LINESIZE:132
$MESSAGE:'Compiling DOSNAME'
$SUBTITLE: '    FUNCTION DOSNAME'
$PAGE
$NOTRUNCATE
      FUNCTION DosName(FNAME)
C
C   "DosName" returns .TRUE. if the character string "FNAME" is one of the
C   DOS reserved names (or special MS Fortran device names), see code below.
C   Otherwise, it returns .FALSE..  The name is checked without any included
C   path or extension first.  If it is found to be a DOS reserved name then
C   the path and extension are removed from "FNAME" to help INQUIRE and OPEN
C   recognize the special names.  If the base name is not one of the DOS
C   reserved names then it is checked against the additional MS Fortran reserved
C   names.  If the base name matches one of the Fortran reserved name and there
C   is no extension or path "DosName"  will be returned .TRUE..
C
C   NOTE:  "FNAME" may be modified by "DosName" so the function call should NOT
C   be made with a constant for "FNAME".
C
      INTEGER LSTCHAR,FSTCHAR
      LOGICAL DosName,SimpleName
      CHARACTER FNAME*(*),FNAMEUP*11
C
C   Get the Name without the path specifaction.
C
      LSTCHAR=NONBLNK(FNAME,LEN(FNAME),1)
      FSTCHAR=SCAN(FNAME,'\,:',.TRUE.)
      IDOT=INDEX(FNAME,'.')
      IF(IDOT .GT. 0)LSTCHAR=IDOT-1
      IF(FSTCHAR .EQ. 0)THEN
         IF(IDOT .EQ. 0)SimpleName=.TRUE.
         FSTCHAR=1
      ELSE
         SimpleName=.FALSE.
         FSTCHAR=FSTCHAR+1
      ENDIF
C
C   Check for match with DOS reserved names with path and extension removed.
C
      FNAMEUP=FNAME(FSTCHAR:LSTCHAR)
      CALL UPPER(FNAMEUP,0)
      DosName=(FNAMEUP .EQ. 'AUX'  .OR. FNAMEUP .EQ. 'COM1' .OR.
     $         FNAMEUP .EQ. 'COM2' .OR. FNAMEUP .EQ. 'CON'  .OR.
     $         FNAMEUP .EQ. 'PRN'  .OR. FNAMEUP .EQ. 'LPT1' .OR.
     $         FNAMEUP .EQ. 'LPT2' .OR. FNAMEUP .EQ. 'LPT3' .OR.
     $         FNAMEUP .EQ. 'NUL')
C
C   If reserved name was found change FNAME to help INQUIRE and OPEN recognize
C   that fact.  If not, check the name with three names reserved by MS FORTRAN,
C   returning true in this case only if no path or extension is present.
C
      IF(DosName)THEN
         FNAME(1:)=FNAME(FSTCHAR:LSTCHAR)
      ELSE
         Dosname=(FNAMEUP .EQ. 'ERR'  .OR. FNAMEUP .EQ. 'LINE' .OR.
     $            FNAMEUP .EQ. 'USER') .AND. SimpleName
      ENDIF
      RETURN
      END
```

```
$PAGESIZE:80
$LINESIZE:132
$MESSAGE:'Compiling DRWCURV'
$SUBTITLE: '    SUBROUTINE DRWCURV'
$PAGE
$NOTRUNCATE
      SUBROUTINE DRWCURV(X,Y,NX,XSIZE,YSIZE,BOXDIR,XMIN,YMIN,
     $ XSCALE,YSCALE,JGRAPH)
C
C   LAST MODIFIED  2/20/89.  BY R. WILSON.
C   *************  5/05/89.  BY R. WILSON.
C        1) Added capability of rotating the graph "BOX".
C        2) The rotation is indicated by the array "BOXDIR" which contains the
C           direction cosines of the box X-axis with the plotter X-axis.
C
C   "DRWCURV" connects the points whose XY coordinates are contained in the
C   arrays "X" and "Y.  The curve is drawn within a box.  Any part of the curve
C   falling outside the box is not drawn.
C
C   "X" and "Y" are real arrays containing the coordinates, in user units, of
C   the points to be connected by a straight line segments.  They are
C   dimensioned "NX".  They are inputs to "DRWCURV".
C
C   "NX" is the size of the coordinate arrays "X" and "Y".  "NX" is an input to
C   "DRWCURV".
C
C   "XSIZE" and "YSIZE" are the X and Y dimensions, respectively, of the box
C   which is to contain the curve.  They are assumed to be in inches.  "XSIZE"
C   and "YSIZE" are inputs to "DRWCURV".
C
C   "BOXDIR" is a real array which contains the direction cosines of the X-axis
C   of the box with the physical plotter axes.  "BOXDIR(1) holds the direction
C   cosine w.r.t. the plotter X-axis.  "BOXDIR(2) holds the direction cosine
C   w.r.t. the plotter Y-axis.  "BOXDIR" is used to rotate the box to any
C   angle.  The angle of the box, in degrees, is stored in "BOXDIR(3)" since
C   the plotter does not have a Z-axis.
C
C   "XMIN" and "YMIN" are the X and Y values, in user coordinates, which are
C   desired at the origin of the box containing the curve.  They are inputs to
C   "DRWCURV".
C
C   "XSCALE" and "YSCALE" are the X and Y scale factors, respectively, used in
C   converting the user coordinates to inches on the plot.  They are inputs to
C   "DRWCURV".
C
C   "JGRAPH" is an integer indicating the number of the current curve being
C   plotted in the box.  If "JGRAPH" is negative then the "JGRAPH"-TH centered
C   symbol will be plotted approximately every inch in arc length along the
C   current curve.  "JGRAPH" is an input to "DRWCURV".
C
      REAL X(NX),Y(NX),BOXDIR(3)
      LOGICAL OUTSIDE
C
C   The following statement functions are used in windowing the plot data if
C   "XMAX, XMIN, YMAX, YMIN" do not include all of the plot data.
C
      XINTSCT(RX,RY,RXL,RYL,RYMAX)=((RX-RXL)/(RY-RYL))*
     $  (RYMAX-RYL)+RXL
      YINTSCT(RX,RY,RXL,RYL,RXMAX)=((RY-RYL)/(RX-RXL))*
     $  (RXMAX-RXL)+RYL
C
C   Scale data in "X" and "Y" and plot.
C
      OUTSIDE=.TRUE.
      DO 1 I=1,NX,1
         XP=(X(I)-XMIN)*XSCALE
         YP=(Y(I)-YMIN)*YSCALE
         IF(XP.GT.XSIZE .OR. XP.LT.0.0 .OR.
     $      YP.GT.YSIZE .OR. YP.LT.0.0 .OR. I.EQ.1)THEN
```

```
C
C         The current point is outside the box.
C
            IF(.NOT.(OUTSIDE))THEN
C
C             The previous point was inside the box.  Linearly interpolate to
C             find the intersection of the curve leaving the box with the box
C             boundary.
C
               XPNEW=XP
               YPNEW=YP
               IF(XPNEW .GT. XSIZE)THEN
                  YPNEW=YINTSCT(XPNEW,YPNEW,XPLAST,YPLAST,XSIZE)
                  XPNEW=XSIZE
               ELSEIF(XPNEW .LT. 0.0)THEN
                  YPNEW=YINTSCT(XPNEW,YPNEW,XPLAST,YPLAST,0.0)
                  XPNEW=0.0
               ENDIF
               IF(YPNEW .GT. YSIZE)THEN
                  XPNEW=XINTSCT(XPNEW,YPNEW,XPLAST,YPLAST,YSIZE)
                  YPNEW=YSIZE
               ELSEIF(YPNEW .LT. 0.0)THEN
                  XPNEW=XINTSCT(XPNEW,YPNEW,XPLAST,YPLAST,0.0)
                  YPNEW=0.0
               ENDIF
               XPAGE=XPNEW*BOXDIR(1)-YPNEW*BOXDIR(2)
               YPAGE=XPNEW*BOXDIR(2)+YPNEW*BOXDIR(1)
               CALL PLTT(XPAGE,YPAGE,2)
            ENDIF
            IPEN=3
            OUTSIDE=.TRUE.
            SLENGTH=0.0
         ELSE
C
C         The current point is inside the box.
C
            IF(OUTSIDE)THEN
C
C             The previous point was outside the box.  Linearly interpolate to
C             find the intersection of the curve entering the box with the box
C             boundary.
C
               IF(XPLAST .GT. XSIZE)THEN
                  YPLAST=YINTSCT(XP,YP,XPLAST,YPLAST,XSIZE)
                  XPLAST=XSIZE
               ELSEIF(XPLAST .LT. 0.0)THEN
                  YPLAST=YINTSCT(XP,YP,XPLAST,YPLAST,0.0)
                  XPLAST=0.0
               ENDIF
               IF(YPLAST .GT. YSIZE)THEN
                  XPLAST=XINTSCT(XP,YP,XPLAST,YPLAST,YSIZE)
                  YPLAST=YSIZE
               ELSEIF(YPLAST .LT. 0.0)THEN
                  XPLAST=XINTSCT(XP,YP,XPLAST,YPLAST,0.0)
                  YPLAST=0.0
               ENDIF
               XPAGE=XPLAST*BOXDIR(1)-YPLAST*BOXDIR(2)
               YPAGE=XPLAST*BOXDIR(2)+YPLAST*BOXDIR(1)
               CALL PLTT(XPAGE,YPAGE,3)
            ENDIF
            IPEN=2
            OUTSIDE=.FALSE.
            SLENGTH=SLENGTH+CABS(CMPLX((XP-XPLAST),(YP-YPLAST)))
         ENDIF
         XPAGE=XP*BOXDIR(1)-YP*BOXDIR(2)
         YPAGE=XP*BOXDIR(2)+YP*BOXDIR(1)
         CALL PLTT(XPAGE,YPAGE,IPEN)
         XPLAST=XP
         YPLAST=YP
```

```
C
C     If "JGRAPH" .LT. 0 plot a centered symbol every 1 inch in arc length.
C
        IF(JGRAPH.LT.0 .AND. SLENGTH.GE.1.0)THEN
           CALL SYMBOL(XPage,YPage,0.1,ABS(JGRAPH),0.0,-1)
           SLENGTH=0.0
        ENDIF
    1 CONTINUE
      RETURN
      END
```

```
$PAGESIZE:80
$LINESIZE:132
$MESSAGE:'Compiling ENG'
$SUBTITLE: '    SUBROUTINE ENG'
$PAGE
$NOTRUNCATE
      SUBROUTINE ENG(X,Y,DIV,IEXP)
C
C   LAST MODIFIED 3/03/85.  BY E. B. JOY
C
C   "ENG" CALCULATES "Y", "DIV" AND "IEXP" SUCH THAT "X"="Y"*"DIV" AND
C   DIV=10**IEXP, WHERE "IEXP" IS SOME MULTILPLE OF 3 AND 1.0 <= ABS("Y")
C   <  1000.  (ACTUALLY 0.001 <= ABS("Y") < 1000.  CHANGE .LT.  TO .LE.  IN
C   "IF(IX.LT.0) IEXP=IEXP-3" TO MAKE THE FIRST INEQUALITY VALID).
C
      IF(X .NE. 0.0)THEN
         IX=INT(ALOG10(ABS(X)))
         IEXP=(IX/3)*3
         IF(IX .LT. 0) IEXP=IEXP-3
         DIV=10.0**IEXP
         Y=X/DIV
      ELSE
         Y=0.0
         IEXP=0
         DIV=1.0
      ENDIF
      RETURN
      END
```

```
$PAGESIZE:80
$LINESIZE:132
$MESSAGE:'Compiling GRAPH'
$SUBTITLE: '    SUBROUTINE GRAPH'
$PAGE
$NOTRUNCATE
      SUBROUTINE GRAPH(X,Y,NX,NXMAX,NOPT,XSIZE,YSIZE,BOXDIR,XMAX,XMIN,
     $  YMAX,YMIN,DELTAX,DELTAY,AXISLBL,NGRAPH,NSPLINE)
C
C    LAST MODIFIED  9/21/87.  BY R. WILSON.
C    *************  1/09/89.  BY R. WILSON.
C         1) MOVED PROTECTION FROM FINDING "YMAX"="YMIN" IN THE DATA TO
C            "AXISXY".
C    *************  2/21/89.  BY R. WILSON.
C         1) NOW USE CALL TO "DRWCURV" TO DRAW EACH CURVE.
C    *************  5/05/89.  BY R. WILSON.
C         1) ADDED CAPABILITY OF ROTATING THE GRAPH "BOX".
C         2) THE ROTATION IS INDICATED BY THE ARRAY "BOXDIR" WHICH CONTAINS THE
C            DIRECTION COSINES OF THE BOX X-AXIS WITH THE PLOTTER AXES.
C
C    THIS IS A GENERAL PURPOSE X-Y PLOT ROUTINE.  IT IS CAPABLE OF PLOTTING
C    LINEAR-LINEAR, SEMILOG(X LINEAR OR Y LINEAR), OR LOG-LOG.
C
C    DESCRIPTION OF PARAMETERS
C
C    "X" AND "Y" ARE ARRAYS CONTAINING THE ABSCISSA AND ORDINATE OF THE CURVE TO
C    BE PLOTTED.  "X" AND "Y" SHOULD BE DIMENSIONED X(NX), Y(NX).
C
C    "NX" IS THE NUMBER OF POINTS IN X AND Y.
C
C    "NOPT" INDICATES HOW THE AXES ARE TO BE LABELED.  SEE "AXISXY" FOR DETAILS.
C
C    "XSIZE" AND "YSIZE" ARE THE MAXIMUM X AND Y DIMENSION OF THE PLOT IN
C    INCHES.
C
C    "BOXDIR" IS A REAL ARRAY WHICH CONTAINS THE DIRECTION COSINES OF THE X-AXIS
C    OF THE BOX WITH THE PHYSICAL PLOTTER AXES.  "BOXDIR(1) HOLDS THE DIRECTION
C    COSINE W.R.T.  THE PLOTTER X-AXIS.  "BOXDIR(2) HOLDS THE DIRECTION COSINE
C    W.R.T.  THE PLOTTER Y-AXIS.  "BOXDIR" IS USED TO ROTATE THE BOX TO ANY
C    ANGLE.  THE ANGLE OF THE BOX, IN DEGREES, IS STORED IN "BOXDIR(3)" SINCE
C    THE PLOTTER DOES NOT HAVE A Z-AXIS.  "GRAPH" ASSUMES THAT THE BOX ORIGIN
C    HAS BEEN SET FROM THE CALLING ROUTINE AND THAT NONE OF THE BOX FALLS
C    OUTSIDE THE PLOTTER MINIMUM LIMITS.  NO CHECK IS DONE FOR THIS.  IF THE
C    ORIGIN HAS NOT BEEN SET CORRECTLY AND PART OF THE BOX DOES FALL OUTSIDE THE
C    PLOTTER MINIMUM LIMITS PROBABLY PARTS OF THE CONTAINED GRAPH WILL NOT BE
C    PLOTTED CORRECTLY SINCE THE CALCOMP ROUTINES TEND TO LOSE THEIR FIX ON THE
C    ORIGIN WHEN THIS HAPPENS.
C
C    "XMAX", "XMIN", "YMAX", "YMIN" ARE THE MAXIMUMS AND MINIMUMS TO BE PLOTTED
C    ON THE RESPECTIVE AXES.  THE MINIMUMS ARE PLOTTED AT THE ORIGIN AND THE
C    MAXIMUMS ARE PLOTTED "XSIZE" AND "YSIZE" INCHES FROM THE RESPECTIVE
C    ORIGINS.  THE MAXIMUMS AND/OR MINIMUMS NEED NOT BE THE ACTUAL MAXIMUMS OR
C    MINIMUMS OF THE DATA.  ALSO "XMAX" DOESN'T NEED TO BE .GT.  "XMIN" AND
C    "YMAX" DOESN'T NEED TO BE .GT.  "YMIN".  IN ADDITION IF THE AXIS MAXIMUM
C    AND MINIMUM ARE EQUAL TO EACH OTHER WHEN "GRAPH" IS CALLED THEN "GRAPH"
C    SCANS THE ARRAY OF VALUES FOR THAT AXIS AND FINDS THE MAXIMUM AND MINIMUM
C    OF THE DATA.  THE DELTA VALUE FOR THAT AXIS IS THEN SET TO ZERO SO THAT
C    WHEN "TPLOT" IS CALLED THE PROGRAM WILL DETERMINE THE TYPE OF SCALING
C    NECESSARY FOR THAT AXIS.
C
C    "DELTAX" AND "DELTAY" ARE USED IN DETERMINING THE TYPE OF AXIS SCALING
C    DESIRED.  IF A DELTA VALUE IS GREATER THAN ZERO THAT AXIS WILL BE SCALED
C    LINEARLY AND THE DELTA VALUE WILL BE USED AS THE TIC MARK INTERVAL.  IF A
C    DELTA VALUE IS LESS THAN ZERO, THEN AFTER A CHECK BY "TPLOT" TO MAKE SURE
C    ZERO DOESN'T OCCUR ON THE AXIS ANYWHERE, THE AXIS IS SCALED LOGARITHMICAL-
C    LY.  IF A DELTA VALUE IS ZERO THAN "TPLOT" WILL DETERMINE THE TYPE OF
C    SCALING FOR THAT AXIS BASED ON THE RATIO OF MAX TO MIN ON THAT AXIS (SEE
C    "TPLOT" FOR DETAILS).  IF "TPLOT" DETERMINES A LINEAR AXIS IS 'POSSIBLE'
C    THEN THE DELTA VALUE REMAINS ZERO SO THAT THE CALL TO "LINAXIS" WILL
```

```
C   PRODUCE LINEAR AUTO-SCALING.  IF "TPLOT" DETERMINES THAT A LOG AXIS IS
C   NECESSARY THE DELTA VALUE IS RETURNED AS -1 FROM "TPLOT" SO THAT A LOG AXIS
C   WILL BE PRODUCED.
C
C   "AXISLBL" IS A ONE DIMENSIONAL CHARACTER ARRAY CONTAINING THE USER SUPPLIED
C   LABEL FOR THE X AND Y AXES.  "AXISLBL(1)" IS USED FOR THE X- AXIS AND
C   "AXISLBL(2)" IS USED FOR THE Y-AXIS.
C
C   "NGRAPH" IS THE NUMBER OF PLOTS WITHIN THIS BOX.  IF ABS(NGRAPH) IS GREATER
C   THAN 1 MORE THAN ONE GRAPH WILL BE PLOTTED ON THE SAME SET OF AXES.  IF
C   NGRAPH .LT.  0 THEN THE INDIVIDUAL GRAPHS WILL BE LABELED WITH CENTERED
C   SYMBOLS EVERY 1 INCH IN ARC-LENGTH ALONG THE GRAPH.
C
C   NOTES:   THE MAIN PROGRAM SHOULD CONTAIN STATEMENTS WHICH
C            INITIALIZE THE PLOT SOFTWARE, MOVE THE ORIGIN
C            UP AND TO THE RIGHT THEN TERMINATE THE PLOT SOFTWARE
C            WHEN ALL DESIRED PLOTS ARE FINISHED. E.G.,
C
C            CALL PLOTS(IBUF,512,NTAPE,PENCODE)
C            CALL PLOT(2.0,2.0,-3)
C                  .
C                PROGRAM
C                  .
C            CALL PLOT(0.0,0.0,999)
C
      PARAMETER (NSPMAX=1024)
      REAL X(NXMAX),Y(NXMAX),XSPLINE(NSPMAX),YSPLINE(NSPMAX),BOXDIR(3)
      CHARACTER AXISLBL(2)*(*)
C
C   IF "XMAX" .EQ.  "XMIN" FIND MAX/MIN OF DATA IN "X(I)" AND USE AS "XMAX" AND
C   "XMIN".  SET "DELTAX" TO 0.0 TO FORCE THE PROGRAM TO DETERMINE THE TYPE OF
C   SCALING ON THE X-AXIS.
C
      IF(XMAX .EQ. XMIN)THEN
         CALL MAXMIN(X,NX,1,NX,XMAX,XMIN,NX,1,1,1)
         DELTAX=0.0
      ENDIF
      CALL TPLOT(XMAX,XMIN,100.0,DELTAX)
C
C   IF "YMAX" .EQ.  "YMIN" FIND MAX/MIN OF DATA IN Y(I) AND USE AS "YMAX" AND
C   "YMIN".  IF MORE THAN 1 PLOT IS TO BE MADE IN THIS BOX USE ALL OF THE
C   Y-DATA FOR THIS BOX IN FINDING THE MAX/MIN OF THE DATA; STORE THESE Y
C   VALUES IN A SCRATCH FILE AND READ THE Y DATA FROM FROM THIS SCRATCH FILE
C   WHEN PLOTTING.  SET "DELTAY"=0.0 TO FORCE THE PROGRAM TO DETERMINE THE TYPE
C   OF SCALING.
C
      IYUNIT=5
      IF(YMAX .EQ. YMIN)THEN
         CALL MAXMIN(Y,NX,1,NX,YMAX,YMIN,NX,1,1,1)
         DELTAY=0.0
         IF(ABS(NGRAPH) .GT. 1)THEN
            OPEN(UNIT=2,STATUS='SCRATCH')
C     WRITE(2,*)NX
C     WRITE(2,*)(X(I),I=1,NX,1)
            WRITE(2,*)(Y(I),I=1,NX,1)
            IYUNIT=2
            DO 1 JGRAPH=2,ABS(NGRAPH),1
C     READ(5,*)NX
C     READ(5,*)(X(I),I=1,NX,1)
               READ(5,*)(Y(I),I=1,NX,1)
C     WRITE(2,*)NX
C     WRITE(2,*)(X(I),I=1,NX,1)
               WRITE(2,*)(Y(I),I=1,NX,1)
               CALL MAXMIN(Y,NX,1,NX,YMAXP,YMINP,NX,1,1,1)
               YMAX=AMAX1(YMAX,YMAXP)
               YMIN=AMIN1(YMIN,YMINP)
    1       CONTINUE
            REWIND(2)
C     READ(2,*)NX
```

```
C     READ(2,*)(X(I),I=1,NX,1)
            READ(2,*)(Y(I),I=1,NX,1)
         ENDIF
      ENDIF
      CALL TPLOT(YMAX,YMIN,100.0,DELTAY)
C
C   DRAW BOX, ANNOTATE AXES, INSURE XSCALE AND YSCALE HAVE CORRECT SIGN, AND
C   CONVERT AXIS MAX/MIN TO LOGARITHMS IF NECESSARY
C
      CALL AXISXY(NOPT,XSIZE,YSIZE,BOXDIR,XMAX,XMIN,YMAX,YMIN,
     $  DELTAX,DELTAY,AXISLBL,XSCALE,YSCALE)
      XSCALE=SIGN(XSCALE,(XMAX-XMIN))
      YSCALE=SIGN(YSCALE,(YMAX-YMIN))
      IF(DELTAX .LT. 0.0)THEN
         XMAX=ALOG10(ABS(XMAX))
         XMIN=ALOG10(ABS(XMIN))
         CALL LOGCON(X,NX)
      ENDIF
      IF(DELTAY .LT. 0.0)THEN
         YMAX=ALOG10(ABS(YMAX))
         YMIN=ALOG10(ABS(YMIN))
      ENDIF
C
C   PLOT THE CURVES IN THIS BOX.
C
      DO 3 JGRAPH=1,ABS(NGRAPH),1
         IF(JGRAPH .GT. 1)READ(IYUNIT,*)(Y(I),I=1,NX,1)
C     IF(JGRAPH .GT. 1)THEN
C        READ(IYUNIT,*)NX
C        READ(IYUNIT,*)(X(I),I=1,NX,1)
C        READ(IYUNIT,*)(Y(I),I=1,NX,1)
C     ENDIF
         IF(DELTAY .LT. 0.0)CALL LOGCON(Y,NX)
         CALL SPLINE(X,Y,NX,XSPLINE,YSPLINE,NSPMAX,NSP,NSPLINE,JGRAPH)
         CALL DRWCURV(XSPLINE,YSPLINE,NSP,XSIZE,YSIZE,BOXDIR,XMIN,YMIN,
     $      XSCALE,YSCALE,SIGN(JGRAPH,NGRAPH))
    3 CONTINUE
      IF(IYUNIT .NE. 5)CLOSE(IYUNIT)
      RETURN
      END
```

```
$PAGESIZE:80
$LINESIZE:132
$MESSAGE:'Compiling INTRPL'
$SUBTITLE: '    SUBROUTINE INTRPL'
$PAGE
$NOTRUNCATE
      SUBROUTINE INTRPL(X,Y,NX,XSPLINE,YSPLINE,NXSPLIN)
C
C   LAST MODIFIED  7/05/87.  BY R. WILSON.
C   ************** 12/01/88.  BY R. WILSON.
C        1) MOVED ERROR MESSAGES INTO FORMAT STATEMENTS
C        2) ADDED LOGICAL VARIABLE "PRGSTOP" SO THAT ALL ERROR CONDITIONS
C           WOULD BE REPORTED AT ONCE.
C   **************  6/29/89.  BY R. WILSON.
C        1) UPDATED THE ERROR MESSAGE FORMAT TO BE CONSISTENT WITH THOSE IN
C           OTHER LIBRARY ROUTINES.
C
C   "INTRPL" INTERPOLATES FROM VALUES OF THE FUNCTION GIVEN AS ORDINATES OF
C   INPUT DATA POINTS IN AN X-Y PLANE AND FOR A GIVEN SET OF X VALUES
C   (ABSCISSAS), THE VALUES OF A SINGLE-VALUED FUNCTION Y=Y(X).  NOTE:  THIS
C   IS A MODIFIED VERSION OF THE ROUTINE DESCRIBED BY H.  AKIMA IMPLEMENTING
C   ALGORITHM 433 FROM THE COMM.  OF THE ACM.  IMSL REFERENCES THIS WHEN
C   DISCUSSING IT'S ROUTINE "IQHSCU" AND CALLS THIS PROCEDURE QUASI-CUBIC
C   HERMITE INTERPOLATION.  "INTRPL" IS A LITTLE MORE GENERAL THAN THE IMSL
C   ROUTINE AND IT ALSO COMPUTES THE VALUES OF THE 'SPLINE'.  THE IMSL ROUTINE
C   REQUIRES A CALL TO ANOTHER SUBPROGRAM TO DO THIS.
C
C   DEFINITION OF PARAMETERS
C
C   "X AND "Y ARE THE INPUT ARRAYS CONTAINING THE INPUT DATA POINTS.  "X"
C   CONTAINS THE ABSCISSAS AND "Y" CONTAINS THE ORDINATES.  THE "X(I)" MUST BE
C   IN ASCENDING ORDER.  A CHECK IS MADE FOR THIS AND THE PROGRAM TERMINATES
C   IF NOT TRUE.
C
C   "NX" IS THE NUMBER OF INPUT DATA POINTS AND THE DIMENSION OF THE "X" AND
C   "Y" ARRAYS.
C
C   "XSPLINE" IS THE INPUT ARRAY CONTAINING THE ABSCISSA VALUES AT WHICH THE
C   INTERPOLATED ORDINATE VALUES ARE DESIRED.  THEY NEED NOT BE ORDERED BUT
C   THE INTERPOLATION RUNS FASTER IF THEY ARE.
C
C   "YSPLINE" IS THE OUTPUT ARRAY CONTAINING THE INTERPOLATED ORDINATE VALUES
C   CORRESPONDING THE ABSCISSA VALUES IN "XPSLINE".
C
C   "NXSPLIN" IS THE NUMBER OF DESIRED INTERPOLATED POINTS AND THE DIMENSION
C   OF "XSPLINE" AND "YSPLINE".  NXSPLIN MUST BE .GE.  1.  A CHECK IS MADE FOR
C   THIS AND THE PROGRAM TERMINATES IF NOT TRUE.
C
      REAL X(NX),Y(NX),XSPLINE(NXSPLIN),YSPLINE(NXSPLIN)
      REAL M1,M2,M3,M4,M5
      LOGICAL PRGSTOP
      EQUIVALENCE (P0,X3),(Q0,Y3),(Q1,T3)
C
C   IT APPEARS THAT THE FOLLOWING EQUIVALENCE STATEMENTS ARE USED JUST TO SAVE
C   MEMORY SPACE.  (commented out here to prevent problems with using
C   $STORAGE:2 on Microsoft compiler command)
C      EQUIVALENCE (XSPLINK,DX),(IMIN,X2,A1,M1),(IMAX,X5,A5,M5)
C      EQUIVALENCE (J,SW,SA),(Y2,W2,W4,Q2),(Y5,W3,Q3)
C   using these EQUIVALENCE statements instead.
      EQUIVALENCE (XSPLINK,DX),(X2,A1),(X5,A5),(IMAX,M5)
      EQUIVALENCE (SW,SA),(Y2,W2,W4,Q2),(Y5,W3,Q3)
C
C   CHECK FOR CORRECT INPUTS.
C
      NXM1=NX-1
      NXP1=NX+1
      PRGSTOP=.FALSE.
      IF(NX .LT. 2)THEN
         WRITE(*,100)NX
```

```
 100     FORMAT(1X,'***** ERROR- program terminated in INTRPL.'/
     $     5X,'NX(= ',I5,') NOT .GE. 2.')
         PRGSTOP=.TRUE.
      ENDIF
      IF(NXSPLIN .LE. 0)THEN
         WRITE(*,101)NXSPLIN
 101     FORMAT(1X,'***** ERROR- program terminated in INTRPL.'/
     $     7X,'NXSPLIN(= ',I5,') .LE. 0.')
         PRGSTOP=.TRUE.
      ENDIF
      IF(PRGSTOP)STOP
C
C   CHECK THAT ABSCISSA VALUES ARE IN ASCENDING ORDER.
C
      DO 11 I=2,NX,1
         IF(X(I-1) .GE. X(I))THEN
            WRITE(*,102)I-1,X(I-1),I,X(I)
 102        FORMAT(1X,'***** ERROR - program terminated in INTRPL.'/
     $        7X,2('X(',I5,' )= ',1PG14.7,:,' .LE. '))
            STOP
         ENDIF
   11 CONTINUE
C
C   PERFORM SPLINE INTERPOLATION.
C
      IPV=0
      DO 80 K=1,NXSPLIN,1
         XSPLINK=XSPLINE(K)
C
C      LOCATE THE DESIRED POINT.
C
         IF(NX .EQ. 2)THEN
            I=2
         ELSEIF(XSPLINK .GE. X(NX))THEN
            I=NXP1
         ELSEIF(XSPLINK .LT. X(1))THEN
            I=1
         ELSE
            IMIN=2
            IMAX=NX
   21       I=(IMIN+IMAX)/2
            IF(XSPLINK .LT. X(I))THEN
               IMAX=I
            ELSE
               IMIN=I+1
            ENDIF
            IF(IMAX .GT. IMIN) GO TO 21
            I=IMAX
         ENDIF
C
C      CHECK IF I=IPV.
C
         IF(I .NE. IPV)THEN
            IPV=I
C
C         PICK UP NECESSARY X AND Y VALUES AND ESTIMATE THEM IF NECESSARY.
C
            J=I
            IF(J .EQ. 1)J=2
            IF(J .EQ. NXP1)J=NX
            X3=X(J-1)
            Y3=Y(J-1)
            X4=X(J)
            Y4=Y(J)
            A3=X4-X3
            M3=(Y4-Y3)/A3
            IF(NX .NE. 2)THEN
               IF(J .NE. 2)THEN
                  X2=X(J-2)
```

```
                  Y2=Y(J-2)
                  A2=X3-X2
                  M2=(Y3-Y2)/A2
               ENDIF
               IF(J .NE. NX)THEN
                  X5=X(J+1)
                  Y5=Y(J+1)
                  A4=X5-X4
                  M4=(Y5-Y4)/A4
                  IF(J .EQ. 2)M2=M3+M3-M4
               ELSE
                  M4=M3+M3-M2
               ENDIF
            ELSE
               M2=M3
               M4=M3
            ENDIF
            IF(J .GT. 3)THEN
               A1=X2-X(J-3)
               M1=(Y2-Y(J-3))/A1
            ELSE
               M1=M2+M2-M3
            ENDIF
            IF(J .LT. NXM1)THEN
               A5=X(J+2)-X5
               M5=(Y(J+2)-Y5)/A5
            ELSE
               M5=M4+M4-M3
            ENDIF
C
C         NUMERICAL DIFFERENTIATION.
C
            IF(I .NE. NXP1)THEN
               W2=ABS(M4-M3)
               W3=ABS(M2-M1)
               SW=W2+W3
               IF(SW .EQ. 0.0)THEN
                  W2=0.5
                  W3=0.5
                  SW=1.0
               ENDIF
               T3=(W2*M2+W3*M3)/SW
            ENDIF
            IF(I .NE. 1)THEN
               W3=ABS(M5-M4)
               W4=ABS(M3-M2)
               SW=W3+W4
               IF(SW .EQ. 0.0)THEN
                  W3=0.5
                  W4=0.5
                  SW=1.0
               ENDIF
               T4=(W3*M3+W4*M4)/SW
            ENDIF
            IF(I .EQ. NXP1)THEN
               T3=T4
               SA=A2+A3
               T4=0.5*(M4+M5-A2*(A2-A3)*(M2-M3)/(SA*SA))
               X3=X4
               Y3=Y4
               A3=A2
               M3=M4
            ENDIF
            IF(I .EQ. 1)THEN
               T4=T3
               SA=A3+A4
               T3=0.5*(M1+M2-A4*(A3-A4)*(M3-M4)/(SA*SA))
               X3=X3-A4
               Y3=Y3-M2*A4
```

```
                 A3=A4
                 M3=M2
              ENDIF
C
C            DETERMINE THE COEFFICIENTS.
C
   60         Q2=(2.0*(M3-T3)+M3-T4)/A3
              Q3=(-M3-M3+T3+T4)/(A3*A3)
           ENDIF
C
C        COMPUTE THE POLYNOMIAL.
C
           DX=XSPLINK-P0
           YSPLINE(K)=Q0+DX*(Q1+DX*(Q2+DX*Q3))
   80 CONTINUE
      RETURN
      END
```

```
$PAGESIZE:80
$LINESIZE:132
$MESSAGE:'Compiling LINAXIS'
$SUBTITLE: '    SUBROUTINE LINAXIS'
$PAGE
$NOTRUNCATE
      SUBROUTINE LINAXIS(XMAX,XMIN,DELTAX,HEIGHT,TICSIZE,
     $  ANGLE,XSIZE,XORIGIN,IAXIS,XSCALE,YLPRIME)
C
C   LAST MODIFIED  3/26/87.  BY R. WILSON.
C   ************* 12/05/88.  BY R. WILSON.
C        1) REMOVED "CHARTYP" FROM ARGUMENT LIST
C        2) "LINAXIS" NOW DRAWS THE AXIS LINE
C   *************  1/10/89.  BY R. WILSON.
C        1) NO LONGER SAVE SIGN OF "DELTAX" SINCE "DELTAX" IS NON-NEGATIVE
C           (POSITIVE OR ZERO) UPON ENTRY AND WANT IT POSITIVE UPON EXIT.
C
C   "LINAXIS" ANNOTATES THE AXIS WITH A LINEAR SCALE.
C
      REAL XORIGIN(2)
      INTEGER NUMDECR
      CHARACTER CENGX*80,IAXIS*8
      LOGICAL LBLAXIS,LBLTIC
      COMMON/LINAXC/MAXDIG
C
C   "INTRND(X,DEL)" IS A STATEMENT FUNCTION INSURING THAT THE INT FUNCTION
C   IS PERFORMED ON "X" ROUNDED TO APPROXIMATELY ALOG10(DEL).
C
      INTRND(X,DEL)=INT(X+SIGN(DEL,X))
C
C   IF "DELTAX"= 0.0 FIND A VALUE FOR "DELTAX" AND NEW VALUES FOR "XMAX" AND
C   "XMIN".  MAKE SURE "DELTAX" HAS THE SAME SIGN AS ("XMAX"-"XMIN").  "DELTAX"
C   WILL BE MADE POSTIVE BEFORE RETURNING.  COMPUTE SCALE FACTOR FOR THIS AXIS
C   (THIS WILL BE MADE POSITIVE BEFORE RETURNING).  COMPUTE SINE AND COSINE OF
C   "ANGLE" FOR ROTATION COMPUTAITON.
C
      IF(DELTAX .EQ. 0.0)THEN
         CALL SCALE1(XMAX,XMIN,.TRUE.,5,XMAXP,XMINP,DELTAX)
         XMAX=XMAXP
         XMIN=XMINP
      ENDIF
      DELTAX=SIGN(DELTAX,(XMAX-XMIN))
      XSCALE=XSIZE/(XMAX-XMIN)
      COSANGL=COSD(ANGLE)
      SINANGL=SIND(ANGLE)
C
C   DETERMINE IF TICMARKS SHOULD BE PLOTTED.  DETERMINE IF NUMBERS SHOULD
C   BE PLOTTED; IF SO INITIALIZE REQ'D VARIABLES, AND COMPUTE FACTORS FOR
C   ENGINEERING NOTATION.
C
      LBLTIC=.TRUE.
      IF(ABS(TICSIZE) .LE. 0.005)LBLTIC=.FALSE.
      IF(IAXIS .EQ. 'NOLABEL')THEN
         LBLAXIS=.FALSE.
      ELSE
         LBLAXIS=.TRUE.
         READ(IAXIS(5:6),'(I2)')NSIGN
         READ(IAXIS(7:8),'(I2)')NSIDE
         YSPACE=NSIDE*(0.8*HEIGHT)
         IF(IAXIS(1:4) .EQ. 'PARL')THEN
            ANGNUM=ANGLE+((1-NSIGN)/2.0)*180.0
            IF(NSIDE .NE. NSIGN)THEN
               YPRIME=YSPACE+NSIDE*HEIGHT
               YLPRIME=YPRIME
            ELSE
               YPRIME=YSPACE
               YLPRIME=YSPACE+NSIDE*HEIGHT
            ENDIF
         ELSEIF(IAXIS(1:4) .EQ. 'PERP')THEN
```

```
            ANGNUM=ANGLE-NSIGN*90.0
            YLPRIME=0.0
         ENDIF
C
C      DETERMINE THE FACTOR, DIVX, BY WHICH ALL NUMBERS APPEARING ON THE
C      AXIS MUST BE MULTIPLIED TO OBTAIN THEIR TRUE VALUE
C
         IF(XMAX.EQ.0.0)THEN
            CALL ENG(XMIN,ENGXM,DIVX,IEXPX)
         ELSEIF(XMIN.EQ.0.0)THEN
            CALL ENG(XMAX,ENGMX,DIVX,IEXPX)
         ELSE
            CALL ENG(XMAX,ENGMX,DIVMX,IEXPMX)
            CALL ENG(XMIN,ENGXM,DIVXM,IEXPXM)
            DIVX=AMAX1(DIVMX,DIVXM)
         ENDIF
         ENGDX=DELTAX/DIVX
         ENGMX=XMAX/DIVX
         ENGXM=XMIN/DIVX
C
C      DETERMINE NUMBER OF DIGITS TO RIGHT OF DECIMAL IN THE ENGINEERING
C      FORM OF THE NUMBERS TO PLOTTED ON THE AXIS.
C
         NDEC=MIN(MAX(NUMDECR(ENGDX,MAXDIG),NUMDECR(ENGMX,MAXDIG),
     $     NUMDECR(ENGXM,MAXDIG)),MAXDIG)
      ENDIF
C
C   ANNOTATE AXIS.
C
      NTX=INTRND((XMAX-XMIN)/DELTAX,2.0E-5)
      NTX1=NTX+1
      IF(XMAX .GT. XMIN)THEN
         XTMIN=XMIN
      ELSE
         XTMIN=XMAX-NTX*DELTAX
      ENDIF
C
C   DRAW AXIS LINE.
C
      CALL PLOT(XORIGIN(1),XORIGIN(2),3)
      XPAGE=XSIZE*COSANGL+XORIGIN(1)
      YPAGE=XSIZE*SINANGL+XORIGIN(2)
      CALL PLOT(XPAGE,YPAGE,2)
C
C   ANNOTATE THE AXIS LINE.
C
      DO 7 I=1,NTX1,1
         XT=XTMIN+(I-1)*DELTAX
         XTI=(XT-XMIN)*XSCALE
         IF(LBLAXIS)THEN
C
C         PLOT NUMBER BY TIC MARK.
C
            ENGX=XT/DIVX
            CALL NUMCHAR(ENGX,NDEC,0,CENGX,NCHAR)
            CALL SLENRW(XLEN,HEIGHT,CENGX,NCHAR)
            IF(IAXIS(1:4) .EQ. 'PARL')THEN
               XPRIME=XTI-NSIGN*XLEN/2.0
            ELSEIF(IAXIS(1:4) .EQ. 'PERP')THEN
               XPRIME=XTI-NSIGN*HEIGHT/2.0
               IF(NSIGN .NE. NSIDE)THEN
                  YPRIME=YSPACE
                  IF(ENGX.GE.0.0 .AND. (XMAX.LT.0.0 .OR. XMIN.LT.0.0))
     $              YPRIME=YPRIME+NSIDE*HEIGHT
                  YLPRIME=NSIDE*AMAX1(ABS(YLPRIME),
     $              ABS(YPRIME+NSIDE*XLEN))
               ELSE
                  YPRIME=YSPACE+NSIDE*XLEN
                  YLPRIME=NSIDE*AMAX1(ABS(YLPRIME),
```

```
     $             ABS(YPRIME))
               ENDIF
            ENDIF
            XPAGE=XPRIME*COSANGL-YPRIME*SINANGL+
     $        XORIGIN(1)
            YPAGE=XPRIME*SINANGL+YPRIME*COSANGL+
     $        XORIGIN(2)
            CALL SYMBOL(XPAGE,YPAGE,HEIGHT,CENGX,ANGNUM,NCHAR)
         ENDIF
         IF(XTI.GT. 0.005 .AND. XTI.LT.(XSIZE-0.005) .AND. LBLTIC)THEN
C
C         PLOT TIC MARK.
C
            XPAGE=XTI*COSANGL+XORIGIN(1)
            YPAGE=XTI*SINANGL+XORIGIN(2)
            CALL PLOT(XPAGE,YPAGE,3)
            XPAGE=XPAGE-TICSIZE*SINANGL
            YPAGE=YPAGE+TICSIZE*COSANGL
            CALL PLOT(XPAGE,YPAGE,2)
         ENDIF
    7 CONTINUE
C
C   COMPUTE Y-COORDINATE (BEFORE ROTATION) OF BOTTOM EDGE OF AXIS
C   LETTERING ASSUMING LETTERING IS DESIRED ON SAME SIDE OF AXIS
C   AS THE NUMBERS.
C
      YLPRIME=YLPRIME+YSPACE+((NSIDE-1)/2)*HEIGHT
C
C   MAKE SURE "XSCALE" AND "DELTAX" ARE POSITIVE.
C
      XSCALE=ABS(XSCALE)
      DELTAX=ABS(DELTAX)
      RETURN
      END
```

```
$PAGESIZE:80
$LINESIZE:132
$MESSAGE:'Compiling LOGAXIS'
$SUBTITLE: '    SUBROUTINE LOGAXIS'
$PAGE
$NOTRUNCATE
      SUBROUTINE LOGAXIS(XMAX,XMIN,EHT,HEIGHT,TICSIZE,ANGLE,
     $  XSIZE,XORIGIN,IAXIS,XSCALE,YLPRIME)
C
C   LAST MODIFIED  7/05/87.  BY R. WILSON.
C   ************* 12/05/88.  BY R. WILSON.
C        1) Removed "chartyp" from argument list.
C        2) "LOGAXIS" now draws the axis line.
C   *************  6/29/89.  BY R. WILSON.
C        1) Updated the error message format to be consistent with those in
C           other library routines.
C
C   "LOGAXIS" annotates the axis with a log scale and returns an "XSCALE" for
C   this axis for use when plotting the data.  "XMAX" and "XMIN" may be
C   modified as described below (but not converted to logarithms).
C
      REAL XORIGIN(2),LOGTEN(10)
      CHARACTER CEXP*80,CHARX*21,IAXIS*8,CSYM*3
      EQUIVALENCE (CHARX,CEXP(1:21))
      LOGICAL LBLAXIS,LBLTIC
      DATA LOGTEN/0.0,0.301029995664,0.4771212547197,0.602059991328,
     $ 0.698970004336,0.7781512503836,0.8450980400143,0.9030899869919,
     $ 0.9542425094393,1.0/
C
C   Initialize "CSYM", the character string representation of 10 or -10.
C
      IF(XMAX.GT.0.0 .AND. XMIN.GT.0.0)THEN
         CSYM(1:2)='10'
         NSYM=2
      ELSEIF(XMAX.LT.0.0 .AND. XMIN.LT.0.0)THEN
         CSYM(1:3)='-10'
         NSYM=3
         XMAX=-XMAX
         XMIN=-XMIN
      ELSE
         WRITE(*,100)
  100    FORMAT(1X,'***** ERROR - program termination in LOGAXIS.'/
     $     7X,'MAX and MIN do not have the SAME sign.')
         STOP
      ENDIF
C
C   Determine a modified "XMIN"="K"*10**"NDECMIN" such that the new "XMIN" is
C   a greatest lower bound of the old "xmin".  Also determine a modified
C   "XMAX"="KMAX"*10**"NDECMAX" such that the new "XMAX" is a least upper
C   bound of the old "XMAX".  "K" and "KMAX" are integers between 1 and 9.
C   "NDECMIN" and "NDECMAX" are integers.
C
      WRITE(CHARX,'(1PE21.14)')XMIN
      READ(CHARX(19:21),'(I3)')NDECMIN
      READ(CHARX(1:2),'(I2)')KMIN
      WRITE(CHARX,'(1PE21.14)')XMAX
      READ(CHARX(19:21),'(I3)')NDECMAX
      READ(CHARX(1:2),'(I2)')KMAX
C
C   If "XMAX" .GT.  "XMIN" increase "KMAX" if necessary and make sure "KMAX"
C   and "KMIN" are between 1 and 9 inclusive.  If "XMAX" .LT.  "XMIN" increase
C   "KMIN" if necessary and make sure "KMAX" and "KMIN" are between 2 and 10
C   inclusive.
C
      IF(XMAX .GT. XMIN)THEN
         IF(MOD((XMAX/10.0**NDECMAX),1.0) .GT. 1.0E-14)THEN
            KMAX=KMAX+1
            IF(KMAX .EQ. 10)THEN
               KMAX=1
```

```
               NDECMAX=NDECMAX+1
            ENDIF
         ENDIF
         NDECDEL=1
         KSTOP=9
      ELSE
         IF(MOD((XMIN/10.0**NDECMIN),1.0) .GT. 2.0E-5)KMIN=KMIN+1
         IF(KMIN .EQ. 1)THEN
            KMIN=10
            NDECMIN=NDECMIN-1
         ENDIF
         IF(KMAX .EQ. 1)THEN
            KMAX=10
            NDECMAX=NDECMAX-1
         ENDIF
         NDECDEL=-1
         KSTOP=2
      ENDIF
C
C   Compute new "XMAX" and "XMIN" and convert to logarithms (anti-log's will
C   be taken and signs will be restored before returning).  Compute scale
C   factor for this axis (this will be made positive before returning),
C   compute cosine and sine of "angle" for rotation computation.
C
      XMAX=LOGTEN(KMAX)+NDECMAX
      XMIN=LOGTEN(KMIN)+NDECMIN
      XSCALE=XSIZE/(XMAX-XMIN)
      COSANGL=COSD(ANGLE)
      SINANGL=SIND(ANGLE)
C
C   Determine if ticmarks should be plotted.  Determine if numbers should be
C   plotted; if so initialize req'd variables.
C
      LBLTIC=.TRUE.
      IF(ABS(TICSIZE) .LE. 0.005)LBLTIC=.FALSE.
      IF(IAXIS .EQ. 'NOLABEL')THEN
         LBLAXIS=.FALSE.
      ELSE
         LBLAXIS=.TRUE.
         READ(IAXIS(5:6),'(I2)')NSIGN
         READ(IAXIS(7:8),'(I2)')NSIDE
         YSPACE=NSIDE*(0.8*HEIGHT)
         IF(IAXIS(1:4) .EQ. 'PARL')THEN
            ANGNUM=ANGLE+((1-NSIGN)/2)*180.0
            IF(NSIDE .NE. NSIGN)THEN
               YPRIME=YSPACE+NSIDE*(HEIGHT+EHT/2.0)
               YLPRIME=YPRIME
            ELSE
               YPRIME=YSPACE
               YLPRIME=YSPACE+NSIDE*(HEIGHT+EHT/2.0)
            ENDIF
            YPRIMEE=YPRIME+NSIGN*(HEIGHT-EHT/2.0)
         ELSEIF(IAXIS(1:4) .EQ. 'PERP')THEN
            ANGNUM=ANGLE-NSIGN*90.0
            YLPRIME=0.0
         ENDIF
      ENDIF
C
C   Draw axis line.
C
      CALL PLOT(XORIGIN(1),XORIGIN(2),3)
      XPAGE=XSIZE*COSANGL+XORIGIN(1)
      YPAGE=XSIZE*SINANGL+XORIGIN(2)
      CALL PLOT(XPAGE,YPAGE,2)
C
C   Annotate axis line.
C
      CALL SLENRW(XLEN10,HEIGHT,CSYM,NSYM)
      CALL SLENRW(XLMINUS,EHT,'-',1)
```

```
      KSTART=KMIN
      DO 8 NDEC=NDECMIN,NDECMAX,NDECDEL
         IF((KSTART.EQ.1 .OR. KSTART.EQ.10) .AND. LBLAXIS)THEN
C
C           Plot power of ten below major tick mark.
C
            IF(NDECDEL .EQ. 1)THEN
               XT=NDEC
            ELSE
               XT=NDEC+1
            ENDIF
            XTI=(XT-XMIN)*XSCALE
            CALL NUMCHAR(XT,0,0,CEXP,NCHARLG)
            CALL SLENRW(XLEN,EHT,CEXP,NCHARLG)
            XLEN=XLEN+XLEN10
            IF(IAXIS(1:4) .EQ. 'PARL')THEN
               XPRIME=XTI-NSIGN*XLEN/2.0
               XPRIMEE=XPRIME+NSIGN*(XLEN10+0.01)
            ELSEIF(IAXIS(1:4) .EQ. 'PERP')THEN
               XPRIME=XTI-NSIGN*HEIGHT/2.0
               XPRIMEE=XPRIME+NSIGN*(HEIGHT-EHT/2.0)
               IF(NSIGN .NE. NSIDE)THEN
                  YPRIME=YSPACE
                  YLPRIME=NSIDE*AMAX1(ABS(YLPRIME),
     $               ABS(YPRIME+NSIDE*XLEN))
               ELSE
                  YPRIME=YSPACE+NSIDE*XLEN
                  IF(XT.GE.0.0 .AND. MIN(NDECMAX,NDECMIN).LT.0)
     $               YPRIME=YPRIME+NSIDE*XLMINUS
                  YLPRIME=NSIDE*AMAX1(ABS(YLPRIME),
     $               ABS(YPRIME))
               ENDIF
               YPRIMEE=YPRIME-NSIGN*(XLEN10+0.01)
            ENDIF
            XPAGE=XPRIME*COSANGL-YPRIME*SINANGL+
     $        XORIGIN(1)
            YPAGE=XPRIME*SINANGL+YPRIME*COSANGL+
     $        XORIGIN(2)
            CALL SYMBOL(XPAGE,YPAGE,HEIGHT,CSYM,ANGNUM,NSYM)
            XPAGE=XPRIMEE*COSANGL-YPRIMEE*SINANGL+
     $        XORIGIN(1)
            YPAGE=XPRIMEE*SINANGL+YPRIMEE*COSANGL+
     $        XORIGIN(2)
            CALL SYMBOL(XPAGE,YPAGE,EHT,CEXP,ANGNUM,NCHARLG)
         ENDIF
         IF(NDEC .EQ. NDECMIN)KSTART=KMIN+NDECDEL
         IF(NDEC .EQ. NDECMAX)KSTOP=KMAX-NDECDEL
         IF((NDECDEL*(KSTOP-KSTART)).GE.0 .AND. LBLTIC)THEN
C
C           Plot tick marks.
C
            DO 7 J= KSTART,KSTOP,NDECDEL
               XT=LOGTEN(J)+NDEC
               XTI=(XT-XMIN)*XSCALE
               IF(J.EQ.1 .OR. J.EQ.10)THEN
                  TICMARK=TICSIZE
               ELSE
                  TICMARK=TICSIZE/2.0
               ENDIF
               XPAGE=XTI*COSANGL+XORIGIN(1)
               YPAGE=XTI*SINANGL+XORIGIN(2)
               CALL PLOT(XPAGE,YPAGE,3)
               XPAGE=XPAGE-TICMARK*SINANGL
               YPAGE=YPAGE+TICMARK*COSANGL
               CALL PLOT(XPAGE,YPAGE,2)
    7       CONTINUE
         ENDIF
         IF(NDECDEL .EQ. 1)THEN
            KSTART=1
```

```
         ELSEIF(NDECDEL .EQ. -1)THEN
            KSTART=10
         ENDIF
    8 CONTINUE
C
C   Compute y-coordinate (before rotation) of bottom edge of axis lettering
C   assuming lettering is desired on same side of axis as the numbers.
C
      YLPRIME=YLPRIME+YSPACE+((NSIDE-1)/2)*HEIGHT
C
C   Make sure "XSCALE" is positive.  Set "XMAX" and "XMIN" To modified values
C   and restore old sign.
C
      XSCALE=ABS(XSCALE)
      IF(CSYM(1:1) .EQ. '-')THEN
         CHARX(1:1)='-'
      ELSE
         CHARX(1:1)=' '
      ENDIF
      CHARX(3:18)='.00000000000000E'
      IF(KMAX .EQ. 10)THEN
         CHARX(2:2)='1'
         WRITE(CHARX(19:21),'(I3)')NDECMAX+1
      ELSE
         WRITE(CHARX(2:2),'(I1)')KMAX
         WRITE(CHARX(19:21),'(I3)')NDECMAX
      ENDIF
      READ(CHARX,'(1PE21.14)')XMAX
      IF(KMIN .EQ. 10)THEN
         CHARX(2:2)='1'
         WRITE(CHARX(19:21),'(I3)')NDECMIN+1
      ELSE
         WRITE(CHARX(2:2),'(I1)')KMIN
         WRITE(CHARX(19:21),'(I3)')NDECMIN
      ENDIF
      READ(CHARX,'(1PE21.14)')XMIN
      RETURN
      END
```

```
$PAGESIZE:80
$LINESIZE:132
$MESSAGE:'Compiling LOGCON'
$SUBTITLE: '    SUBROUTINE LOGCON'
$PAGE
$NOTRUNCATE
      SUBROUTINE LOGCON(X,NX)
C
C   LAST MODIFIED 10/07/85.  BY R. WILSON.
C
      PARAMETER (ZERO=1.0E-12)
      REAL X(NX)
      DO 1 I=1,NX,1
         AX=ABS(X(I))
         IF(AX .GT. ZERO)THEN
            X(I)=ALOG10(AX)
         ELSE
            X(I)=-12.0
         ENDIF
    1 CONTINUE
      RETURN
      END
```

```
$PAGESIZE:80
$LINESIZE:132
$MESSAGE:'Compiling MAXMIN'
$SUBTITLE: '    SUBROUTINE MAXMIN'
$PAGE
$NOTRUNCATE
      SUBROUTINE MAXMIN(ARRAY,NX,NY,NXMAX,AMAX,AMIN,IMAX,IMIN,JMAX,JMIN)
C
C   LAST MODIFIED 8/15/86.  BY R. WILSON.
C
      REAL ARRAY(NXMAX,*),AMAX,AMIN
C
C   "MAXMIN" FINDS THE MAXIMUM AND MINIMUM OF THE DATA IN THE REAL ARRAY
C   "ARRAY" FOR THE ARRAY INDICES (I,J) WHERE
C
C              "IMIN" .LE. I .LE. "IMAX"
C              "JMIN" .LE. J .LE. "JMAX"
C
C   MAKE SURE "IMIN", "IMAX", "JMIN", AND "JMAX" ARE WITHIN THE PROPER RANGE.
C
      IMINARA=MIN(IMIN,IMAX)
      IMAXARA=MAX(IMIN,IMAX)
      JMINARA=MIN(JMIN,JMAX)
      JMAXARA=MAX(JMIN,JMAX)
      IMINARA=MIN(MAX(1,IMINARA),NX)
      IMAXARA=MIN(MAX(1,IMAXARA),NX)
      JMINARA=MIN(MAX(1,JMINARA),NY)
      JMAXARA=MIN(MAX(1,JMAXARA),NY)
C
C   FIND MAX AND MIN WITHIN INDICATED REGION OF I AND J.
C
      AMAX=ARRAY(IMINARA,JMINARA)
      AMIN=ARRAY(IMINARA,JMINARA)
      DO 1 J=JMINARA,JMAXARA,1
         DO 1 I=IMINARA,IMAXARA,1
            IF(ARRAY(I,J) .GT. AMAX)AMAX=ARRAY(I,J)
            IF(ARRAY(I,J) .LT. AMIN)AMIN=ARRAY(I,J)
    1 CONTINUE
      RETURN
      END
```

```
$PAGESIZE:80
$LINESIZE:132
$MESSAGE:'Compiling NONBLNK'
$SUBTITLE: '    FUNCTION NONBLNK'
$PAGE
$NOTRUNCATE
      FUNCTION NONBLNK(CSTRING,ISTART,ISTOP)
C
C   LAST MODIFIED 11/18/87.  BY R. WILSON.
C
      INTEGER DELTA,NONBLNK
      CHARACTER CSTRING*(*)
C
C   "NONBLNK" determines the first non-blank character in the character string
C   "CSTRING" if "ISTART" is .LT. "ISTOP".  It determines the last non-blank
C   character in the string if "ISTART" is .GT. "ISTOP".  If no non-blank
C   characters are found "NONBLNK" is set to 0.
C
      DELTA=SIGN(1,(ISTOP-ISTART))
      DO 1 I=ISTART,ISTOP,DELTA
         NONBLNK=I
         IF(CSTRING(I:I) .NE. ' ')RETURN
    1 CONTINUE
      NONBLNK=0
      RETURN
      END
```

```
$PAGESIZE:80
$LINESIZE:132
$MESSAGE:'Compiling NUMCHAR'
$SUBTITLE: '     SUBROUTINE NUMCHAR'
$PAGE
$NOTRUNCATE
      SUBROUTINE NUMCHAR(XNUM,NDECR,OFFSET,CNUM,NCHAR)
C
C   LAST MODIFIED 10/08/87.  BY R. WILSON.
C   *************  3/15/88.  BY R. WILSON.
C        1) Changed second argument name from "NRIGHT" to "NDECR".  This
C           corrected an error which would occur when "NUMCHAR" is called
C           with a 'CONSTANT' second argument which is negative.
C   ************* 12/05/88.  BY R. WILSON.
C        1) Removed "CHARTYP" from argument list.
C   *************  2/05/90.  BY R. WILSON.
C        1) Use ABS("OFFSET") to trap the possibilty that the user may
C           inadvertantly use an "OFFSET" < 0.  Note: in MS Fortran 5.0
C           this does not cause a runtime error, however, the results are
C           incorrect.  Also, a runtime error is generated on CYBER.
C
C   "NUMCHAR" converts the real number "XNUM" to a character string stored
C   in "CNUM".
C
C   ABS("NDECR") denotes the desired number of digits to the right of the
C   decimal point in the character version of the real number "XNUM".  For
C   "NDECR" .GE.  0 "CNUM" will be the character version of "XNUM" with "XNUM"
C   having been rounded to "NDECR" digits to the right of the decimal place.
C   Usually "NDECR" is .GE.  0.  However, if "NDECR" is .LT.  0 then "NUMDECR"
C   is called to determine the number of non-zero digits to be included right
C   of the decimal point in "CNUM".  The absolute value of this negative
C   "NDECR" is used in the call to "NUMDECR" as the maximum allowed number of
C   digits.  Note: as indicated below calling "NUMCHAR" with a constant negative
C   value (the argument in the call is a number or parameter not a variable)
C   for NDECR" will not cause problems.  "NDECR" is an input to "NUMCHAR".
C
C   "OFFSET" is the starting character position in "CNUM" to be used for the
C   string representing the real number "XNUM".  Usually "OFFSET"=0 but in some
C   cases "OFFSET" > 0; in which case the characters "CNUM(1:OFFSET) are not
C   destroyed.  "OFFSET" is an input to "NUMCHAR".
C
C   "NCHAR" is the number of characters stored in "CNUM" upon return from
C   "NUMCHAR".  This value includes the "OFFSET" value input to "NUMCHAR". That
C   is, "NCHAR" is the length of the string in "CNUM". NCHAR" is an output
C   from "NUMCHAR".
C
      CHARACTER CNUM*(*),INTCHAR*18,CFORMAT*80
      INTEGER FSTCHAR,OFFSET,LCNUM
C
C   "INTRND(X,XDEL)" is a statement function insuring that the INT function is
C   performed on X rounded to account for any round off error.
C
      INTRND(X,XDEL)=INT(X+SIGN(XDEL,X))
C
C   Find length of "CNUM".   Make sure "OFFSET" is non-negative.
C
      LCNUM=LEN(CNUM)
      IOFFSET=ABS(OFFSET)
C
C   Find the number of character in "XNUM".  Note:  "NRIGHT" is set equal to
C   the dummy argument "NDECR".  Then "NRIGHT" is used in the remainder of
C   "NUMCHAR" to avoid the problem of changing the input "NDECR" when "NUMCHAR"
C   is called with a constant value for "NDECR" which is negative.
C
      NRIGHT=NDECR
      IF(ABS(XNUM) .GT. 1.0)THEN
         NCHAR=INTRND(ALOG10(ABS(XNUM)),2.0E-5)+1
      ELSE
         NCHAR=1
```

```
      ENDIF
      IF(XNUM .LT. 0.0)NCHAR=NCHAR+1
      IF(NRIGHT .LT. 0)NRIGHT=NUMDECR(XNUM,ABS(NRIGHT))
      NCHAR=NCHAR+NRIGHT+1
C
C   Initialize "CFORMAT".
C
      CFORMAT='(F'
C
C   Convert "NLEFT" and "NRIGHT" to character, load into "CFORMAT"
C
      WRITE(INTCHAR,FMT='(I18)')NCHAR
      FSTCHAR=NONBLNK(INTCHAR,1,18)
      LENGTH=18-FSTCHAR
      CFORMAT(3:LENGTH+3)=INTCHAR(FSTCHAR:18)
      IDOT=LENGTH+4
      CFORMAT(IDOT:IDOT)='.'
      WRITE(INTCHAR,FMT='(I18)')NRIGHT
      FSTCHAR=NONBLNK(INTCHAR,1,18)
      LENGTH=18-FSTCHAR
      CFORMAT(IDOT+1:IDOT+LENGTH+1)=INTCHAR(FSTCHAR:18)
      CFORMAT(IDOT+LENGTH+2:)=')'
C
C   Using format string in "CFORMAT" write "XNUM" to "CNUM("OFFSET"+2:)".
C
      WRITE(CNUM(IOFFSET+1:),CFORMAT(1:IDOT+LENGTH+2))XNUM
C
C   Adjust the string so that a leading "0" is included for "XNUM" < 1.0.
C
      IDOT=INDEX(CNUM(IOFFSET+1:),'.')+IOFFSET
      IF(CNUM(IDOT-2:IDOT) .EQ. ' -.')THEN
         CNUM(IDOT-2:IDOT-1)='-0'
      ELSEIF(CNUM(IDOT-1:IDOT) .EQ. ' .')THEN
         CNUM(IDOT-1:IDOT-1)='0'
      ENDIF
      CALL REMOVE(CNUM(IOFFSET+1:),NCHAR)
      NCHAR=NCHAR+IOFFSET
C
C   If "NRIGHT" is 0 ("XNUM" is a whole number) don't include the decimal
C   point in the string.
C
      IF(NRIGHT .EQ. 0)THEN
         IDOT=INDEX(CNUM(IOFFSET+1:),'.')+IOFFSET
         CNUM(IDOT:)=' '
         NCHAR=IDOT-1
      ENDIF
      RETURN
      END
```

```
$PAGESIZE:80
$LINESIZE:132
$MESSAGE:'Compiling NUMDECR'
$SUBTITLE: '    FUNCTION NUMDECR'
$PAGE
$NOTRUNCATE
      FUNCTION NUMDECR(X,MAXDECR)
C
C   LAST MODIFIED  3/26/87.  BY R. WILSON.
C   *************  2/05/90.  BY R. WILSON.
C        1) Return if fractional part of "X" is zero instead of returning if
C           "X" is zero.
C        2) If the fractional part of "X" is zero, return with "NUMDECR"=0.
C
      INTEGER NUMDECR,MAXDECR
      CHARACTER CHARX*21,FORMAT*8
C
C   "NUMDECR" determines the number of digits, including leading zeroes,
C   to the right of the decimal point in the number "X".  This is performed
C   on "X" rounded to "MAXDECR" significant digits to the right of the
C   decimal point.
C
      NUMDECR=0
      XFRACT=X-INT(X)
      IF(XFRACT .NE. 0.0)THEN
C
C      If the fractional part of "X" is non-zero write this fractional part
C      into a character variable.  When the "XFRACT" is written into "CHARX"
C      the result will contain "MAXDECR" significant figures.  If the
C      fractional part were not used below (i.e.  replace XFRACT with X) then
C      for "X" .gt.  1.0 only (MAXDECR-NLEFT) digits would be reported as
C      existing.  For use with plot routines this would generally not be a
C      problem, but the definition of "MAXDECR" would not be implemented.
C
         NSIGFIG=MIN(MAXDECR,14)
         FORMAT='(E21.'
         WRITE(FORMAT(6:),'(I2,'')'')')NSIGFIG
         WRITE(CHARX,FORMAT)XFRACT
         READ(CHARX(19:21),'(I3)')NUMDECR
         ICHAR1=18-NSIGFIG
C
C      Since only the fractional part is being used the exponent "NUMDECR" is
C      usually .LE.  0.  However, the INT function may cause rounding (for
C      example if X=.999999999999) so the next "IF" statement is used to insure
C      that the character variable is "SCANNED" correctly.
C
         IF(NUMDECR .GE. 0)ICHAR1=ICHAR1+NUMDECR
         NUMDECR=NSIGFIG-NUMDECR
         DO 1 I=17,ICHAR1,-1
            IF(CHARX(I:I) .NE. '0')GO TO 2
            NUMDECR=NUMDECR-1
    1    CONTINUE
    2    CONTINUE
      ENDIF
      RETURN
      END
```

```
$PAGESIZE:80
$LINESIZE:132
$MESSAGE:'Compiling OPNFILE'
$SUBTITLE: '    SUBROUTINE OPNFILE'
$PAGE
$NOTRUNCATE
      SUBROUTINE OPNFILE(IUNIT,IOTYPE,FNAME,NAMLGTH,PROMPT,WHICHARG)
C
C    LAST MODIFIED  5/26/89.  BY R. WILSON.
C    *************  7/26/89.  BY R. WILSON.
C         1) Modified to use new MS Fortran 5.0 routines NARGS, and GETARG
C            which return command line argument.
C         2) Added "WHICHARG" to the argument list.
C    *************  9/06/89.  BY R. WILSON.
C         1) Renamed parameter "MODE" to "IOTYPE".
C         2) Acceptable parameters for "IOTYPE" are now INPUT, OUTPUT, and
C            APPEND.
C         3) If IOTYPE is APPEND the file is opened in APPEND mode, i.e. at
C            the end of information ready for writing.
C    ************* 11/15/89.  BY R. WILSON.
C         1) Re-worded the messages produced by format statements 101 and 102.
C
C    "OPNFILE" Opens a file for INPUT, OUTPUT, or APPEND.
C
C    "IUNIT" is the unit number to open the file on.  It is an INPUT to
C    "OPNFILE".
C
C    "IOTYPE" is a character variable containing the mode to open the file in.
C    It can have the values 'INPUT', 'OUTPUT', or 'APPEND'.  A check is made for
C    the existence of the file.  If the file is to be opened for INPUT and it
C    does not exist an informative message is displayed and the user is asked
C    for a new file name.  If the file is to be opened for OUTPUT and it does
C    exist the user is asked if it is okay to overwrite.  If the file is opened
C    for APPEND and it does not exist an informative message is displayed and
C    the user is asked for a new file name.  If "IOTYPE" is not 'INPUT',
C    'OUTPUT', or 'APPEND', no checks are made for existence of the file and the
C    old file contents, if any, will be destroyed by any write operation.
C
C    "FNAME" is the name of the file to be and which is eventually opened.
C    If "FNAME" is blank then, if "WHICHARG" is .GT. 0, the "WHICHARG" argument
C    on the command line is checked for a file.  If a file name is not found
C    there then the user is prompted for a file name.  "FNAME" is an INPUT to
C    and an OUTPUT from "OPNFILE".
C
C    "NAMLGTH" is the number of characters in "FNAME".  it is an OUTPUT from
C    "OPNFILE".
C
C    "PROMPT" is a character string to be used in the prompt for the file name
C    when "FNAME" is blank, no file is specified in the command tail, or "FNAME"
C    (obtained from the keyboard or the command tail) already exists.  "PROMPT"
C    is an INPUT to "OPNFILE".
C
C    "WHICHARG" is the number of the command line argument to check for a file
C    name if "FNAME" is blank.  If "WHICHARG" is .LE. 0 or there are fewer than
C    "WHICHARG" arguments on the command line (in addition to the executable
C    file name which is argument number 0) the user is prompted for a file name.
C    "WHICHARG" is an INPUT to "OPNFILE".
C

      CHARACTER FNAME*(*),PROMPT*(*),IOTYPE*6,IOTYPE1*6
      INTEGER IUNIT,NAMLGTH,MXNLGTH,NUMARGS,PLGTH,WHICHARG
      INTEGER*4 NARGS
      LOGICAL FEXIST,ISDOS,YESNOTF,DOSNAME
      EXTERNAL YESNOTF,DOSNAME
C
C    Check for a blank file name.  Get length of "PROMPT".
C
      MXNLGTH=LEN(FNAME)
      NAMLGTH=NONBLNK(FNAME,MXNLGTH,1)
```

```
      PLGTH=NONBLNK(PROMPT,LEN(PROMPT),1)
C
C   If the name is blank look in the command tail for the name.
C
      IF(NAMLGTH .EQ. 0)THEN
         NUMARGS=INT2(NARGS())-1
         IF(WHICHARG .GT. 0 .AND. WHICHARG .LE. NUMARGS)THEN
            CALL GETARG(WHICHARG,FNAME,NAMLGTH)
         ELSE
            WRITE(*,100)PROMPT(1:PLGTH)
  100       FORMAT(/1X,A,' '\)
            READ(*,'(A)')FNAME
            NAMLGTH=NONBLNK(FNAME,MXNLGTH,1)
         ENDIF
      ENDIF
C
C  Check for existence of the file if "IOTYPE" is 'INPUT', 'OUTPUT' or
C  'APPEND'.
C
      IOTYPE1=IOTYPE
      CALL UPPER(IOTYPE1,6)
    1 CONTINUE
      ISDOS=DosName(FNAME)
      CALL UPPER(FNAME,NAMLGTH)
      INQUIRE(FILE=FNAME,EXIST=FEXIST)
      IF(IOTYPE1(1:6) .EQ. 'OUTPUT')THEN
C
C   Check if the file name is a DOS reserved name, and check for file
C   existance.
C
         FEXIST=FEXIST .AND. .NOT.(ISDOS)
C
C      Want to only write to file.  If it already exists ask if okay to
C      overwrite.
C
         IF(FEXIST)THEN
            CALL BEEP
            WRITE(*,101)FNAME(1:NAMLGTH)
  101       FORMAT(/1X,'***** WARNING - You requested output to be',
     $        1X,'WRITTEN to the file:'//7X,A//
     $        1X,'which already exists.'/)
            IF( .NOT. (YESNOTF('Is it okay to overwrite')))THEN
               WRITE(*,100)PROMPT(1:PLGTH)
               READ(*,'(A)')FNAME
               NAMLGTH=NONBLNK(FNAME,MXNLGTH,1)
               GO TO 1
            ENDIF
         ENDIF

      ELSEIF(IOTYPE1(1:5) .EQ. 'INPUT')THEN
C
C      Want to only read from file.  If it does not exist ask for name again.
C
         FEXIST=FEXIST .OR. ISDOS
         IF(.NOT. (FEXIST))THEN
            CALL BEEP
            WRITE(*,102)FNAME(1:NAMLGTH)
  102       FORMAT(/1X,'***** ERROR - You requested input to be',
     $        1X,'READ from the file:'//7X,A//
     $        1X,'which does not exist.  Try again.')
            WRITE(*,100)PROMPT(1:PLGTH)
            READ(*,'(A)')FNAME
            NAMLGTH=NONBLNK(FNAME,MXNLGTH,1)
            GO TO 1
         ENDIF
      ENDIF
C
C  OPEN THE FILE.
C
```

```
      IF(IOTYPE1(1:6) .EQ. 'APPEND')THEN
         OPEN(IUNIT,FILE=FNAME,ACCESS='APPEND')
      ELSE
         OPEN(IUNIT,FILE=FNAME)
      ENDIF
C
      RETURN
      END
```

```
$PAGESIZE:80
$LINESIZE:132
$MESSAGE:'Compiling PLTT'
$SUBTITLE: '    SUBROUTINE PLTT'
$PAGE
$NOTRUNCATE
      SUBROUTINE PLTT(X,Y,IPEN)
C
C   LAST MODIFIED 6/13/85.
C
C   PLTT ELIMINATES MOVING PEN FOR HIDDEN LINES.
C
      XLAST=XN
      YLAST=YN
      ILAST=IN
      XN=X
      YN=Y
      IN=IPEN
      IF(IPEN .EQ. 2)THEN
         IF(ILAST .EQ. 2)THEN
            CALL PLOT(X,Y,IPEN)
         ELSEIF(ILAST .EQ. 3)THEN
            CALL PLOT(XLAST,YLAST,ILAST)
            CALL PLOT(X,Y,IPEN)
         ENDIF
      ELSEIF(IPEN .LT. 0)THEN
         CALL PLOT(X,Y,IPEN)
      ENDIF
      RETURN
      END
```

```
$PAGESIZE:80
$LINESIZE:132
$MESSAGE:'Compiling READERR'
$SUBTITLE: '    SUBROUTINE READERR'
$PAGE
$NOTRUNCATE
      SUBROUTINE READERR
C
C   LAST MODIFIED 8/3/89.  BY R. WILSON.
C
C   "READERR" writes an message to the screen indicating IMPROPER input and
C   instructing the user to try again.
C
      CALL BEEP
      WRITE(*,100)
  100 FORMAT(1X,'***** ERROR - Improper input.  Try again!'/)
      RETURN
      END
```

```
$PAGESIZE:80
$LINESIZE:132
$MESSAGE:'Compiling REC2POL'
$SUBTITLE: '    FUNCTION REC2POL'
$PAGE
$NOTRUNCATE
      FUNCTION REC2POL(CRECT,DB)
C
C   LAST MODIFIED 11/13/87. BY R. WILSON
C   *************  6/21/88. BY R. WILSON
C         1) Changed to a function and added to "PLTLIB"
C
      PARAMETER(PI=3.1415926535898,RAD2DEG=180.0/PI,ZERO=1.0E-12)
      COMPLEX CRECT,REC2POL
      LOGICAL DB
C
C   "REC2POL" computes the magnitude and phase of the complex number "CRECT"
C   and stores the magnitude in the real part of "REC2POL" and the phase in
C   the imaginary part of "REC2POL".  If "DB" is TRUE then the dB amplitude
C   will be stored.  The pahse will be in degrees.
C
      AMP=CABS(CRECT)
      IF(AMP .LE. ZERO)THEN
         PHASE=0.0
      ELSE
         PHASE=ATAN2(AIMAG(CRECT),REAL(CRECT))*RAD2DEG
      ENDIF
      IF(DB)THEN
         AMP=20.0*ALOG10(MAX(AMP,ZERO))
      ENDIF
      REC2POL=CMPLX(AMP,PHASE)
      RETURN
      END
```

```
$PAGESIZE:80
$LINESIZE:132
$MESSAGE:'Compiling REMOVE'
$SUBTITLE: '    SUBROUTINE REMOVE'
$PAGE
$NOTRUNCATE
      SUBROUTINE REMOVE(CSTRING,NCHAR)
C
C   LAST MODIFIED  9/19/85.  BY R. WILSON.
C   **************  6/28/88.  BY R. WILSON.
C        1) Corrected error which occurs when attempt is made to add
C           colon on end of "CSTRING" when colon position would be passed
C           end of "CSTRING".
C   ************** 12/05/88.  BY R. WILSON.
C        1) Modified to run using PLOT88 which doesn't stop plotting when
C           a colon is encountered in the string.  So don't want to add the
C           the colon.
C   **************  7/18/89.  BY R. WILSON.
C        1) Corrected error in logic which left the tail of "CSTRING" in the
C           new, "leading-blank-removed" "CSTRING".
C        2) This only shows up if you call "REMOVE" again with the same
C           "CSTRING" multiple times (as is now done in "AXIS3D").
C        3) This error may not show up with some compilers.  I don't recall
C           seeing this error up with the Microsoft compiler for example.
C
C   "REMOVE" removes leading blanks from the character string "CSTRING".  The
C   number of characters in the string "CSTRING" is returned as "NCHAR".
C
      INTEGER FSTCHAR,LSTCHAR,LSTRING
      CHARACTER*(*) CSTRING
C
C   FIND LENGTH OF "CSTRING".
C
      LSTRING=LEN(CSTRING)
C
C   REMOVE LEADING BLANK SPACES.
C
      NCHAR=0
      FSTCHAR=NONBLNK(CSTRING,1,LSTRING)
      IF(FSTCHAR .NE. 0)THEN
         LSTCHAR=NONBLNK(CSTRING,LSTRING,FSTCHAR)
         NCHAR=LSTCHAR-FSTCHAR+1
         IF(FSTCHAR .NE. 1)THEN
            CSTRING(1:)=CSTRING(FSTCHAR:LSTCHAR)
         ENDIF
      ENDIF
      RETURN
      END
```

```
$PAGESIZE:80
$LINESIZE:132
$MESSAGE:'Compiling SCALE1'
$SUBTITLE: '    SUBROUTINE SCALE1'
$PAGE
$NOTRUNCATE
      SUBROUTINE SCALE1(XMAX,XMIN,CONTAIN,NTICSM1,XMAXP,XMINP,DELTAX)
C
C    LAST MODIFIED 3/27/86.  BY R. WILSON
C
C    GIVEN "XMAX", "XMIN"(.NE. "XMAX"), AND "NTICSM1" SCALE1 FINDS A NEW RANGE
C    RANGE ("XMAXP", "XMINP") DIVISIBLE INTO APPROXIMATELY "NTICSM1" LINEAR
C    INTERVALS OF SIZE DELTAX.  IF "CONTAIN" IS TRUE THEN THE INTERVAL
C    ("XMIN","XMAX") WILL BE CONTAINED IN ("XMINP","XMAXP").  THIS IS
C    THE SITUATION NORMALLY USED IN LINEARLY SCALING AN AXIS FOR PLOTTING
C    (LINAXIS).  IF "CONTAIN" IS FALSE THEN ("XMINP","XMAXP") WILL BE
C    CONTAINED IN ("XMIN","XMAX").  THIS WOULD BE THE SITUATION NORMALLY
C    USED IN SCALING THE CONTOUR VALUES IN CONTOUR PLOTS (CONTR).
C
C    IF "DELTAX"=0 ON INPUT THEN AN ADJUSTED "DELTAX" IS FOUND AND USED IN
C    FINDING "XMAXP" AND "XMINP".  OTHERWISE THE "DELTAX" VALUE PASSED ON INPUT
C    IS USED IN FINDING "XMAXP" AND "XMINP".
C
C    "VINT" IS AN ARRAY OF ACCEPTABLE VALUES FOR "DELTAX" (TIMES AN INTEGER
C    POWER OF 10.0).
C
C    "SQR" IS AN ARRAY OF GEOMETRIC MEANS OF ADJACENT VALUES OF "VINT", USED AS
C    BREAK POINTS TO DETERMINE WHICH "VINT" VALUE TO ASSIGN TO "DELTAX."
C
      REAL VINT(4),SQR(3)
      LOGICAL CONTAIN,ORDERED
      DATA VINT/1.,2.,5.,10./
      DATA SQR/1.414214,3.162278,7.071068/
C
C    "DEL" ACCOUNTS FOR COMPUTER ROUND-OFF.  IT SHOULD BE GREATER THAN THE
C    ROUND-OFF EXPECTED FROM A DIVISION  AND FLOAT OPERATION.  IT SHOULD BE
C    LESS THAN THE MINIMUM INCREMENT OF THE PLOTTING DEVICE USED BY THE MAIN
C    PROGRAM IN INCHES DIVIDED BY THE PLOT SIZE IN INCHES TIMES THE NUMBER
C    OF INTERVALS.
C
      DEL=0.00002
C
C    IF "DELTAX" PASSED TO "SCALE1" IS 0.0 FIND A NEW ADJUSTED VALUE  FOR
C    "DELTAX", OTHERWISE USE THE "DELTAX" VALUE PASSED TO "SCALE1" IN
C    DETERMINING THE ADJUSTED "XMAX" AND "XMIN".
C
      IF(DELTAX .EQ. 0)THEN
C
C       FIND APPROXIMATE INTERVAL SIZE.
C
         DELTAX=ABS(XMAX-XMIN)/NTICSM1
         NDIGL=INT(ALOG10(DELTAX))
         IF(DELTAX .LT. 1.)NDIGL=NDIGL-1
C
C       SCALE "DELTAX" INTO A NUMBER BETWEEN 1 AND 10.
C
         SDELTAX=DELTAX/10.0**NDIGL
C
C       USE "SDELTAX" TO FIND CLOSEST "VINT" VALUE.
C
         DO 1 I=1,3,1
            IF(SDELTAX .LT. SQR(I))GO TO 2
    1    CONTINUE
         I=4
C
C       COMPUTE ACTUAL "DELTAX".
C
    2    CONTINUE
         DELTAX=VINT(I)*10.0**NDIGL
```

```
      ENDIF
C
C   FIND NEW" XMAX" AND "XMIN".
C
      ORDERED=(XMAX .GT. XMIN)
      IF(CONTAIN .EQV. ORDERED)THEN
         FM1=XMIN/DELTAX
         FM2=XMAX/DELTAX
      ELSE
         FM1=XMAX/DELTAX
         FM2=XMIN/DELTAX
      ENDIF
      M1=INT(FM1)
      IF(FM1 .LT. 0.0)M1=M1-1
      IF(ABS(FLOAT(M1)+1.0-FM1) .LT. DEL)M1=M1+1
      M2=INT(FM2+1.)
      IF(FM2 .LT. -1.)M2=M2-1
      IF(ABS(FM2+1.0-FLOAT(M2)) .LT. DEL)M2=M2-1
      IF(CONTAIN .EQV. ORDERED)THEN
         XMINP=DELTAX*FLOAT(M1)
         XMAXP=DELTAX*FLOAT(M2)
      ELSE
         XMINP=DELTAX*FLOAT(M2)
         XMAXP=DELTAX*FLOAT(M1)
      ENDIF
      IF(CONTAIN)THEN
C
C   ADJUST LIMITS IF NECESSARY TO ACCOUNT FOR ROUNDOFF.
C
         IF(ORDERED)THEN
            IF((XMINP-XMIN) .GT. DEL)XMINP=XMIN
            IF((XMAX-XMAXP) .GT. DEL)XMAXP=XMAX
         ELSE
            IF((XMIN-XMINP) .GT. DEL)XMINP=XMIN
            IF((XMAXP-XMAX) .GT. DEL)XMAXP=XMAX
         ENDIF
      ENDIF
      RETURN
      END
```

```
$PAGESIZE:80
$LINESIZE:132
$MESSAGE:'Compiling SIND'
$SUBTITLE: '    FUNCTION SIND'
$PAGE
$NOTRUNCATE
      FUNCTION SIND(ANGLE)
C
C   LAST MODIFIED 12/01/88.  BY. R. WILSON.
C
C   "SIND" COMPUTES THE SINE OF THE ANGLE "ANGLE" WHERE "ANGLE" IS IN
C   DEGREES.
C
      PARAMETER(PI=3.1415926535898,DEG2RAD=PI/180.0)
      SIND=SIN(ANGLE*DEG2RAD)
      RETURN
      END
```

```
$PAGESIZE:80
$LINESIZE:132
$MESSAGE:'Compiling SLENRW'
$SUBTITLE: '    SUBROUTINE SLENRW'
$PAGE
$NOTRUNCATE
      SUBROUTINE SLENRW(WIDTH,HEIGHT,TEXT,N)
C
C   LAST MODIFIED  5/13/85.  BY R. WILSON.
C   ************* 12/01/88.  BY R. WILSON.
C        1)  Modified to work with the PLOT88 library
C        2)  To this end changed from calling SLEN to GETWID
C        3)  In the future might want to remove "SLENRW" and change
C            any calls to it in the plot library to calls to "GETWID"
C
      CHARACTER*(*)TEXT
C
C   "WIDTH" is the returned length of the text in the character string
C   "TEXT".  "HEIGHT" is the desired height of the text and "N" is the
C    number of characters in "TEXT" to be considered.
C
      CALL GETWID(HEIGHT,TEXT,N,WIDTH)
      RETURN
      END
```

```
$PAGESIZE:80
$LINESIZE:132
$MESSAGE:'Compiling SPLINE'
$SUBTITLE: '    SUBROUTINE SPLINE'
$PAGE
$NOTRUNCATE
      SUBROUTINE SPLINE(X,Y,NX,XSPLINE,YSPLINE,NXSPMAX,NXSP,
     $ NSPLINE,JGRAPH)
C
C   LAST MODIFIED  7/05/87.  BY R. WILSON.
C   *************  2/02/87.  BY R. WILSON.
C        1) MODIFIED ERROR MESSAGE SO IT PROVIDES MORE INFORMATION.
C   *************  6/29/89.  BY R. WILSON.
C        1) UPDATED THE ERROR MESSAGE FORMAT TO BE CONSISTENT WITH THOSE IN
C           OTHER LIBRARY ROUTINES.
C
C   "SPLINE" TAKES THE DATA IN THE ARRAYS "X" AND "Y" AND LOADS IT INTO
C   THE CORRESPONDING ARRAYS "XSPLINE" AND "YSPLINE".  IF "NSPLINE"=1 THEN
C   'LOCAL' SPLINE SMOOTHING IS PERFORMED.
C
C   DEFINITION OF PARAMETERS.
C
C   FOR THE DEFINITONS OF "X", "Y", "NX", "XSPLINE", AND "YSPLINE" SEE THE
C   LISTING FOR "INTRPL".
C
C   "NXSPMAX" IS THE MAXIMUM NUMBER OF POINTS AT WHICH SMOOTHING IS DESIRED.
C   IT IS ALSO THE DIMENSION OF THE ARRAYS "XSPLINE" AND "YSPLINE".
C
C   "NXSP" IS SET EQUAL TO "NXSPMAX" IF SMOOTHING IS DESIRED. IT IS SET EQUAL
C   TO "NX" IF SMOOTHING IS NOT DESIRED.
C
C   "NSPLINE" CONTROLS THE TYPE OF SMOOTHING PERFORMED AS FOLLOWS:
C         NSPLINE=0    NO SMOOTHING IS PERFORMED
C                =1    'LOCAL' SPLINE SMOOTHING IS PERFORMED
C
C   "JGRAPH" DETERMINES WHETHER THE ABSCISSCA VALUES STORED IN "XSPLINE" ARE
C   RESET.  IF "JGRAPH" .EQ. 1 THEN THEY ARE RESET, IF NOT THEN SMOOTHING IS
C   PERFORMED AT THE OLD ABSCISSA VALUES.
C
      REAL X(NX),Y(NX),XSPLINE(NXSPMAX),YSPLINE(NXSPMAX)
      IF(NX .GT. NXSPMAX)THEN
         WRITE(*,100)NX,NXSPMAX
  100    FORMAT(1X,'***** ERROR- program terminated in SPLINE.'/
     $     7X,'Dimension of arrays X and Y (NX=',I5,') exceeds',
     $     1X,'dimension of arrays'/7X,'XSPLINE and YSPLINE',
     $     1X,'(NXSPMAX=',I5,').')
         STOP
      ENDIF
      IF(NSPLINE .GT. 0)THEN
         IF(JGRAPH .EQ. 1)THEN
C
C          CALCULATE ABSCISSA VALUES AT WHICH SPLINE APPROXIMATIONS
C          ARE DESIRED AND STORE IN XSPLINE.
C
            NXSP=NXSPMAX
            UFAC=(X(NX)-X(1))/(NXSPMAX-1.0)
            DO 2 I=1,NXSP,1
               XSPLINE(I)=(I-1)*UFAC+X(1)
    2       CONTINUE
         ENDIF
C
C       CALCULATE VALUES OF 'LOCAL' SPLINE APPROXIMATION AT THE XSPLINE(I).
C       STORE IN YSPLINE.
C
         CALL INTRPL(X,Y,NX,XSPLINE,YSPLINE,NXSPMAX)
      ELSE
C
C       NO SPLINE APPROXIMATION DESIRED.
C
```

```
         NXSP=NX
         DO 3 I=1,NXSP,1
            XSPLINE(I)=X(I)
            YSPLINE(I)=Y(I)
 3       CONTINUE
      ENDIF
      RETURN
      END
```

```
$MESSAGE:'Compiling TIME'
$SUBTITLE: '   Function TIME'
$PAGE
$NOTRUNCATE
      FUNCTION TIME()
C
C   LAST MODIFIED  8/28/88.  BY R. WILSON.
C   *************  5/25/89.  BY R. WILSON.
C        1) ADDED CODE TO FORMAT THE SECONDS PART OF THE TIME STRING AS
C           '01', '02', ETC., FOR SECONDS .LE. 9.
C
C   "TIME" RETURNS THE CURRENT TIME (AS REPORTED BY THE SYSTEM CLOCK).  MS
C   FORTRAN HAS A LIBRARY ROUTINE CALLED "GETTIM" WHICH IS CALLED BY "TIME".
C   THE RETURNED INTEGER*2 VALUES ARE PACKED INTO A CHARACTER VARIABLE IN CDC
C   CYBER NOS/VE FORMAT AND RETURNED AS "TIME".
C
      INTEGER*2 IHR,IMIN,ISEC,I100TH
      CHARACTER TIME*8
C
C   GET THE CURRENT TIME FROM THE SYSTEM CLOCK.
C
      CALL GETTIM(IHR,IMIN,ISEC,I100TH)
C
C   PACK TIME INTO CHARACTER VARIABLE "TIME" IN CDC NOS FORMAT.
C
      WRITE(TIME,100)IHR,IMIN,ISEC
  100 FORMAT(I2,':',I2,':',I2)
      IF(TIME(4:4) .EQ. ' ')THEN
           TIME(4:4)='0'
      ENDIF
      IF(TIME(7:7) .EQ. ' ')THEN
           TIME(7:7)='0'
      ENDIF
      RETURN
      END
```

```
$PAGESIZE:80
$LINESIZE:132
$MESSAGE:'Compiling TITLE'
$SUBTITLE: '    SUBROUTINE TITLE'
$PAGE
$NOTRUNCATE
      SUBROUTINE TITLE(CTITLE,ANGLE,NSIGN,YLETTER,HEIGHT,XSIZE,XORIGIN)
C
C   LAST MODIFIED  6/01/86.  BY R. WILSON.
C
C   "TITLE" PLOTS THE CHARACTER STRING "CTITLE" PARALLEL TO AN AXIS WHICH
C   STARTS AT "XORIGIN", IS OF LENGTH "XSIZE" AND IS AT AN ANGLE "ANGLE"
C   TO THE HORIZONTAL.  THE CHARACTERS IN THE TITLE WILL BE PLOTTED WITH
C   HEIGHT "HEIGHT", AND THEIR BOTTOM EDGE WILL BE AT "YLETTER" BEFORE
C   ROTATION (TO THE ANGLE "ANGLE").  THE CHARACTERS WILL BE CENTERED
C   ALONG THE AXIS.  ANY LEADING BLANKS IN THE STRING "CTITLE" WILL BE
C   REMOVED BEFORE CENTERING.  IF "NSIGN"= +1 THE CHARACTERS WILL BE
C   PLOTTED PARALLEL TO THE AXIS.  IF "NSIGN"= -1 THE CHARACTERS WILL BE
C   PLOTTED ANTI-PARALLEL TO THE AXIS.
C
      REAL XORIGIN(2)
      CHARACTER CTITLE*(*)
C
C   REMOVE LEADING BLANKS AND CENTER TITLE ALONG AXIS.
C
      CALL REMOVE(CTITLE,NCHAR)
      IF(NCHAR .GT. 0)THEN
         CALL SLENRW(XTITLE,HEIGHT,CTITLE,NCHAR)
         XLETTER=(XSIZE-NSIGN*XTITLE)/2.0
         XPAGE=XLETTER*COSD(ANGLE)-YLETTER*SIND(ANGLE)+
     $      XORIGIN(1)
         YPAGE=XLETTER*SIND(ANGLE)+YLETTER*COSD(ANGLE)+
     $      XORIGIN(2)
         ANGTITL=ANGLE+((1-NSIGN)/2.0)*180.0
         CALL SYMBOL(XPAGE,YPAGE,HEIGHT,CTITLE,ANGTITL,NCHAR)
      ENDIF
      RETURN
      END
```

```
$PAGESIZE:80
$LINESIZE:132
$MESSAGE:'Compiling TPLOT'
$SUBTITLE: '    SUBROUTINE TPLOT'
$PAGE
$NOTRUNCATE
      SUBROUTINE TPLOT(XMAX,XMIN,TMAX,DELTAX)
C
C   LAST MODIFIED 10/13/85.  BY R. WILSON.
C
C   DETERMINE TYPE OF AXIS SCALING TO BE USED IN PLOTTING AXIS, LINEAR OR
C   LOGARITHMIC.
C
C   "DELTAX" IS AN INPUT-OUTPUT VARIABLE.  FOR "DELTAX" .LT. 0.0 THE USER IS
C   REQUESTING A LOGARITHMIC AXIS. "TPLOT" CHECKS TO MAKE SURE THIS IS POSSIBLE
C   (I.E. THAT 0 WILL NOT APPEAR SOMEWHERE ON THE AXIS BETWEEN "XMAX" AND
C   "XMIN") AND IF A LOG AXIS IS POSSIBLE A NEGATIVE "DELTAX" VALUE WILL BE
C   RETURNED.  IF A LOGARITHMIC AXIS IS NOT POSSIBLE "DELTAX"=0.0 WILL BE
C   RETURNED SO THAT "AXISXY" WILL PLOT AN AUTO-SCALED LINEAR AXIS.  FOR
C   "DELTAX"=0.0 THE USER IS ASKING "TPLOT" TO DETERMINE THE TYPE OF AXIS
C   SCALING TO USE.  IF THE RATIO, "TLOG" (DISCUSSED BELOW) IS .LE. "TMAX" THEN
C   A "DELTAX"=0.0 IS RETURNED AND AN AUTO-SCALED LINEAR AXIS IS OBTAINED.  IF
C   "TLOG" IS .GT. "TMAX" A NEGATIVE "DELTAX" IS RETURNED AND A LOGARITHMIC
C   AXIS IS OBTAINED.  FOR "DELTAX" .GT. 0.0 THE USER IS ASKING FOR A LINEAR
C   PLOT AND "TPLOT" RETURNS THE POSITIVE "DELTAX" VALUE TO GIVE IT TO HIM.
C   IN THIS CASE "LINAXIS" WILL NOT AUTO-SCALE.
C
C   "TLOG" IS DEFINED THE WAY IT IS IN ORDER TO ALLOW THE POSSIBILITY (HOWEVER
C   REMOTE) THAT THE USER WANTS THE MAXIMUM VALUE ON THE AXIS LEFT OF (BELOW)
C   THE MINIMUM VALUE ON THE AXIS.
C
      IF(DELTAX .LT. 0.0)THEN
         IF((XMAX*XMIN) .LE. 0.0)DELTAX=0.0
      ELSEIF(DELTAX .EQ. 0.0)THEN
         ACTMIN=MIN(XMAX,XMIN)
         IF(ACTMIN .EQ. 0.0)RETURN
         ACTMAX=MAX(XMAX,XMIN)
         TLOG=ACTMAX/ACTMIN
         IF(TLOG.LE.0.0)RETURN
         IF(TLOG.LE.TMAX)RETURN
         DELTAX=-1.0
      ENDIF
      RETURN
      END
```

```
$PAGESIZE:80
$LINESIZE:132
$MESSAGE:'Compiling UPPER'
$SUBTITLE: '    SUBROUTINE UPPER'
$PAGE
$NOTRUNCATE
      SUBROUTINE UPPER(CSTRING,NUM2UP)
C
C   LAST MODIFIED  8/02/88.  BY R. WILSON.
C   *************  1/13/89.  BY R. WILSON.
C        1) CONVERSION FROM THE FIRST CHARACTER TO THE LAST NON-BLANK
C           CHARACTER IN "CSTRING" IS NOW PERFORMED BOTH FOR "NUM2UP" <= 0 AND
C           "NUM2UP" > "LSTCHAR".
C
C   "UPPER" CONVERTS THE FIRST "NUM2UP" CHARACTERS IN THE STRING "CSTRING" TO
C   UPPER CASE.  IF "NUM2UP" <= 0 OR > "LSTCHAR" THEN THE ROUTINE WILL SCAN
C   THROUGH THE STRING FROM THE FIRST CHARACTER TO THE LAST NON-BLANK
C   CHARACTER CONVERTING TO UPPER CASE IF NECESSARY.
C
      INTEGER LSTCHAR,MXNCHAR
      CHARACTER*(*) CSTRING
C
C   DETERMINE LENGTH OF "CSTRING".
C
      MXNCHAR=LEN(CSTRING)
      LSTCHAR=NONBLNK(CSTRING,MXNCHAR,1)
C
C   IF "CSTRING" IS NOT ALL BLANKS SCAN THROUGH "CSTRING" REPLACING LOWER
C   CASE CHARACTERS WITH THEIR UPPER CASE EQUIVALENTS.  IF "CSTRING" IS JUST
C   BLANK MAKE IT A SINGLE BLANK CHARACTER.
C
      IF(LSTCHAR .GT. 0)THEN
         IF(NUM2UP.LE.0 .OR. NUM2UP.GT.LSTCHAR)THEN
            NCHAR=LSTCHAR
         ELSE
            NCHAR=NUM2UP
         ENDIF
         DO 1 I=1,NCHAR,1
            ICOLSEQ=ICHAR(CSTRING(I:I))
            IF(ICOLSEQ.GE.97 .AND. ICOLSEQ .LE. 122)THEN
               CSTRING(I:I)=CHAR(ICOLSEQ-32)
            ENDIF
    1    CONTINUE
      ELSE
         CSTRING=' '
      ENDIF
      RETURN
      END
```

```
$PAGESIZE:80
$LINESIZE:132
$MESSAGE:'Compiling YesNoTF'
$SUBTITLE: '    FUNCTION YesNoTF'
$PAGE
$NOTRUNCATE
      FUNCTION YesNoTF(PROMPT)
C
C   LAST MODIFIED  3/20/89. BY R. WILSON.
C
      CHARACTER PROMPT*(*),YESNO*1
      LOGICAL YesNoTF
C
C   PROMPTS FOR A "Y"es OR "N"o ANSWER TO THE STANDARD OUTPUT DEVICE USING
C   THE PROMPT STRING "PROMPT" AND CONVERTS A "Y"es ANSWER TO .TRUE. AND
C   A "N"o ANSWER TO .FALSE..
C
    1 CONTINUE
      WRITE(*,100)PROMPT
  100 FORMAT(1X,A,' (y or n)? '\)
      READ(*,'(A1)')YESNO
      CALL UPPER(YESNO,1)
      IF(YESNO .EQ. 'N')THEN
         YesNoTF=.FALSE.
      ELSEIF(YESNO .EQ. 'Y')THEN
         YesNoTF=.TRUE.
      ELSE
         GO TO 1
      ENDIF
      RETURN
      END
```

```
C     *********************************************************************
C                          ANTENNA LAYOUT PROGRAM
C     *********************************************************************
C     MAIN PROGRAM AND LAYOUT-RELATED SUBROUTINES.
C     NETWORK DESIGN AND INDUCTANCE/CAPACITANCE CALCULATIONS IN
C     MODULE NETS.
C     *********************************************************************
C
      REAL LANT,LAMBDA
      DIMENSION XO(20,20),YO(20,20),EL1(20,20)
      DIMENSION EL2(20,20),EL3(20,20),MTYPE(20,20)
      DIMENSION XP(30),YP(30),XX(20,20),YY(20,20)
      DIMENSION XH(400),YH(400),ZH(400),MH(400)
      CHARACTER*1 FEEDLOC
      LOGICAL RFLAG
C
      COMMON /WINSIZ/WX1,WY1,WX2,WY2,FACT
C
      DATA XMAX/10.5/,YMAX/7.3/
      DATA WNORM/0.15/
      DATA YCLR/-3E10/
      DATA LTOP/1/,LBOT/2/,LBRD/3/
      DATA PI/3.141596/
      DATA FTHICK/0.0014/
C
C
      OPEN(UNIT=11,FORM='FORMATTED',FILE=' ',STATUS='OLD')
C     READ FREQUENCY IN GHz
      READ(11,*)GFREQ
      LAMBDA=3E10/(1E9*GFREQ*2.54)
C     READ XSIZE AND YSIZE OF THE ANTENNA LAYOUT.
      READ(11,*)XDIM,YDIM
C
C     READ DIELECTRIC CONSTANT AND THICKNESS
      READ(11,*)EPSR,THICK
      THICK=THICK*1E3
C     READ THE TIP-TO-TIP ANTENNA LENGTH, ANTENNA WIDTH AND ANTENNA TILT.
      READ(11,*)LANT,WANT,TILT
      IF(TILT.GE.0.)THEN
         ANGLE=PI/2.-TILT*PI/180.
      ELSE
         ANGLE=-PI/2.-TILT*PI/180.
      ENDIF
C     WRITE(0,*)'TILT, ANGLE: ',TILT,ANGLE

      IF(ABS(ANGLE).GE.1E-6)THEN
         IROT=1
      ELSE
         IROT=0
      ENDIF
C
C     READ FEED NETWORK LOCATION.
      READ(11,100)FEEDLOC
         IF(FEEDLOC.EQ.'B')THEN
            ITOP=0
            ISWAP=0
            ILOC=1
         ELSE IF(FEEDLOC.EQ.'T')THEN
            ITOP=1
            ISWAP=0
            ILOC=2
         ELSE IF(FEEDLOC.EQ.'L')THEN
            ITOP=0
            ISWAP=1
            ILOC=1
         ELSE IF(FEEDLOC.EQ.'R')THEN
            ITOP=1
            ISWAP=1
            ILOC=2
```

```
         ELSE
            ITOP=0
            ISWAP=0
            ILOC=1
         ENDIF
C
C
   100 FORMAT(A1)
C  READ THE NUMBER OF COLUMNS AND ROWS IN THE ARRAY.
      READ(11,*)NCOL,NROW
C
      IF(NCOL.GT.20.OR.NROW.GT.20)THEN
         WRITE(0,*)'Storage limits antenna to 20 rows or columns.'
         STOP
      ENDIF
C
C  READ DATA IN ONE ROW AT A TIME
C
      DO 1 M=1,NROW
      DO 1 N=1,NCOL
         READ(11,*)XO(M,N),YO(M,N),CURR,PHASE,ZMAG,ZANGL
    1 CONTINUE
C
C  READ THE MATCHING NETWORK DATA
C
      DO 2 M=1,NROW
      DO 2 N=1,NCOL
         READ(11,*)NC,NR,MTYPE(M,N),EL1(M,N),EL2(M,N),EL3(M,N)
    2 CONTINUE
C
      JJ=0
      DO 10 N=1,NCOL
      DO 10 M=1,NROW
      JJ=JJ+1
      MH(JJ)=MTYPE(M,N)
      XH(JJ)=XO(M,N)
   10 YH(JJ)=YO(M,N)
C  10 WRITE(0,*)XH(JJ),YH(JJ)
C
      IF(ISWAP.EQ.1)THEN
C
         NZZZ=NROW
         NROW=NCOL
         NCOL=NZZZ
C
         JJ=0
         DO 11 M=1,NROW
         DO 11 N=1,NCOL
         JJ=JJ+1
         MTYPE(M,N)=MH(JJ)
         XO(M,N)=YH(JJ)
   11    YO(M,N)=XH(JJ)
C
         NZZZ=NROW
         NROW=NCOL
         NCOL=NZZZ
      JJ=0
      DO 12 N=1,NCOL
      DO 12 M=1,NROW
         JJ=JJ+1
         XH(JJ)=EL1(M,N)
         YH(JJ)=EL2(M,N)
   12    ZH(JJ)=EL3(M,N)
C  12 WRITE(0,*)XH(JJ),YH(JJ),ZH(JJ)
C
         NZZZ=NROW
         NROW=NCOL
         NCOL=NZZZ
      JJ=0
```

```
      DO 13 M=1,NROW
      DO 13 N=1,NCOL
         JJ=JJ+1
         EL1(M,N)=YH(JJ)
         EL2(M,N)=XH(JJ)
   13    EL3(M,N)=ZH(JJ)
C  13 WRITE(0,*)EL1(M,N),EL2(M,N),EL3(M,N)
C
         XTEMP=XDIM
         XDIM=YDIM
         YDIM=XTEMP
         ANGLE=-TILT*PI/180.
C
      ENDIF
C
C  READ THE DATA FOR THE ARRAY MATCHING NETWORK.
C     READ(11,*)MT,ET1,ET2,ET3
C
      IF(GFREQ.GE.1.0)THEN
         WFEED=.0625
      ELSE
         WFEED=.125
      ENDIF
C
      LANT=LANT/2.
  777 WRITE(0,500)
  500 FORMAT(//1H ,'Antenna Layout Program:  Version 1/11/90'//1H ,
     & 'Select Desired Procedure'/1H ,
     & 5X,'1-Automatically Scale Plot to EGA Screen'/1H ,
     & 5X,'2-Automatically Scale Plot to HP-7570  Plotter on COM1'/1H ,
     & 5X,'3-Automatically Scale Plot to HP-7585  Plotter on COM2'/1H ,
     & 5X,'4-Automatically Scale Plot to HP-LJet Plotter on LPT1'/1H ,
     & 5X,'5-Plot With Scale = 1 to HP-7585 Plotter on COM2'/1H ,
     & 5X,'6-Specify Plotter Port, Type, Scale, and Plot Window'//)
C
      WRITE(0,501)
  501 FORMAT(1H ,'Enter Number to Indicate Choice: '\)
      READ(0,*)ICHOICE
      IF(ICHOICE.GT.6.OR.ICHOICE.LT.1)GO TO 777
C
      IF(ICHOICE.EQ.1)THEN
         IOPORT=97
         MODEL=97
         WX1=0.
         WY1=0.
         WX2=XDIM
         WY2=YDIM
         FACT=1.
      ENDIF
C
      IF(ICHOICE.EQ.2)THEN
         IOPORT=9600
         MODEL=20
         WX1=0.
         WY1=0.
         WX2=XDIM
         WY2=YDIM
         FACT=AMIN1(XMAX/XDIM,YMAX/YDIM)
      ENDIF
C
      IF(ICHOICE.EQ.3)THEN
         IOPORT=9650
         MODEL=85
         WX1=0.
         WY1=0.
         WX2=XDIM
         WY2=YDIM
         FACT=AMIN1(64./XDIM,34./YDIM)
         IF(FACT.GT.1.)FACT=1.
```

```
      ENDIF
C
      IF(ICHOICE.EQ.4)THEN
         IOPORT=0
         MODEL=62
         WX1=0.
         WY1=0.
         WX2=XDIM
         WY2=YDIM
         FACT=AMIN1(9.9/XDIM,7.9/YDIM)
      ENDIF
C
      IF(ICHOICE.EQ.5)THEN
         IOPORT=9650
         MODEL=85
         WX1=0.
         WY1=0.
         WX2=XDIM
         WY2=YDIM
         FACT=1.
      ENDIF
C
      IF(ICHOICE.EQ.6)THEN
      WRITE(0,222)
  222 FORMAT(1H ,'Enter IOPORT, MODEL, SCALE FACTOR: '\)
      READ(0,*)IOPORT,MODEL,FACT
      WRITE(0,223)
  223 FORMAT(1H ,'Enter WINDOW Values: '\)
      READ(0,*)WX1,WY1,WX2,WY2
      ENDIF
C
C
C  IF ONLY ONE ROW PRESENT, SET DISTANCE FROM THE FEED POINT TO
C  BE ONE-HALF WAVELENGTH.
C
      IF(NROW.GT.1)THEN
         YDIST=YO(2,1)-YO(1,1)
      ELSE
         YDIST=LAMBDA/2.
      ENDIF
C
C  SET UP THE PLOTTING
      CALL PLOTS(0,IOPORT,MODEL)
C     WRITE(0,*)'WX1,WY1,WX2,WY2: ',WX1,WY1,WX2,WY2
C
      BX1=WX1
      BX2=WX2
      BY1=WY1
      BY2=WY2
C
      CALL WINDOW(BX1,BY1,BX2,BY2)
C
C
C     WRITE(0,*)'Scale Factor: ',FACT
      CALL FACTOR(FACT)
C
C     WRITE(0,*)'WX1,WY1,WX2,WY2: ',WX1,WY1,WX2,WY2
C
      CALL PLOT(WX1,WY1,3)
      CALL PLOT(WX1,WY2,2)
      CALL PLOT(WX2,WY2,2)
      CALL PLOT(WX2,WY1,2)
      CALL PLOT(WX1,WY1,2)
C
C  DRAW THE ANTENNAS AND FEED MATCHING NETWORKS
C  SEND A MESSAGE TO THE OPERATOR
      WRITE(0,300)
  300 FORMAT(/1H ,'Constructing Antennas and Feed Structure ....'//)
C
```

```
      CALL FEEDPNT(NROW,NCOL,XO,YO,MTYPE,EL1,EL2,EL3,
     &   LANT,WANT,WFEED,THICK,EPSR,FTHICK,IROT,ANGLE,
     &   YCLR,LAMBDA,ILOC,M)
C
C     WRITE(0,*)'EXIT FEEDPNT; YCLR: ',YCLR
C
C DRAW THE FEED LINE EXTENSIONS
      CALL FEEDLIN(NROW,NCOL,XO,YO,WFEED,IROT,ANGLE,YCLR,
     &             ITOP,YDIST)
C
C IF MORE THAN ONE ROW, CONSTRUCT MEANDERED TRANSMISSION LINES BETWEEN
C ANTENNAS.
C
      IF(NROW.GT.1)THEN
         CALL MEANDER(XP,YP,XP1,YP1,NPOINT,YCLR,YDIST,ANGLE,
     &                WNORM,LAMBDA,EPSR,.FALSE.)
C
         CALL PATH(XO,YO,NROW,NCOL,XP,YP,XP1,YP1,NPOINT,WFEED,LTOP)
         CALL PATH(XO,YO,NROW,NCOL,XP,YP,XP1,YP1,NPOINT,WFEED,LBOT)
      ELSE
         XP1=-YCLR*SIN(ANGLE)
         YP1=YCLR*COS(ANGLE)
      ENDIF
C
C CONSTRUCT MEANDERED TRANSMISSION LINES FROM THE FEED POINTS.
C THIS MEANDER LINE IS ALWAYS VERTICAL AND MAY BE A DIFFERENT
C LENGTH FROM THAT CONNECTING THE ANTENNAS.  SUBROUTINE
C MEANDER MUST BE CALLED.
C
C     WRITE(0,*)'CALL MEANDER; YCLR: ',YCLR
      CALL MEANDER(XP,YP,XP2,YP2,NPOINT,YCLR,YDIST+YCLR-YP1,0.,
     &             WNORM,LAMBDA,EPSR,.FALSE.)
C
      IF(ITOP.NE.1)THEN
         DO 5 N=1,NCOL
         XX(1,N)=XO(1,N)-XP1
    5    YY(1,N)=YO(1,1)-YDIST+YCLR
      ELSE
         DO 7 N=1,NCOL
         XX(1,N)=XO(1,N)+XP1
    7    YY(1,N)=YO(NROW,1)+YP1
      ENDIF
C
      CALL PATH(XX,YY,2,NCOL,XP,YP,0.0,0.0,NPOINT,WFEED,LTOP)
      CALL PATH(XX,YY,2,NCOL,XP,YP,0.0,0.0,NPOINT,WFEED,LBOT)
C
C DRAW THE T-SHAPED FEED LINE EXTENSIONS
C
      CALL TEE(XO,YO,XP1,YP1,NROW,NCOL,YDIST,
     &         YCLR,WFEED,LTOP,ITOP)
      CALL TEE(XO,YO,XP1,YP1,NROW,NCOL,YDIST,
     &         YCLR,WFEED,LBOT,ITOP)
C
C DRAW THE MEANDERED LINE SECTIONS CONNECTING THE ANTENNA COLUMNS
C SUBROUTINE MEANDER MUST BE CALLED TO CONSTRUCT A LINE OF DIFFERENT
C PROPERTIES.
C
      IF(NCOL.GT.1)THEN
         XDIST=XO(1,2)-XO(1,1)
C
         CALL MEANDER(XP,YP,XTMP,YTMP,NPOINT,YCLR,XDIST,0.,
     &                WNORM,LAMBDA,EPSR,.TRUE.)
C
         IF(ITOP.NE.1)THEN
            DO 4 N=1,NCOL-1
            XX(1,N)=XO(1,N)+YCLR-XP1
    4       YY(1,N)=YO(1,1)-YDIST
C   4       WRITE(0,*)'XX,YY: ',XX(1,N),YY(1,N)
         ELSE
```

```
              DO 6 N=1,NCOL-1
              XX(1,N)=XO(1,N)+YCLR+XP1
    6         YY(1,N)=YO(NROW,1)+YDIST
C   6         WRITE(0,*)'XX,YY: ',XX(1,N),YY(1,N)
           ENDIF
C
           CALL PATH(XX,YY,2,NCOL-1,XP,YP,0.0,0.0,NPOINT,WFEED,LTOP)
           CALL PATH(XX,YY,2,NCOL-1,XP,YP,0.0,0.0,NPOINT,WFEED,LBOT)
C
        ENDIF
C
C  ALL COORDINATES HAVE BEEN GENERATED. -- PLOT THE ANTENNA
C
        CALL DRAWANT
C
        WRITE(0,610)FACT
  610 FORMAT(1H ,6X,'Scale Factor of Plot: ',F8.3)
C
        IF(ISWAP.EQ.1)THEN
        WRITE(0,600)
  600 FORMAT(//1H ,'Feed networks located on left or right side ',/1H ,
     &             'will plot with x- and y-coordinates interchanged.'/)
        ENDIF
C
C
        STOP
        END
C
C
        SUBROUTINE FEEDPNT(NROW,NCOL,XO,YO,MTYPE,EL1,EL2,EL3,
     &    LANT,WANT,WFEED,THICK,EPSR,FTHICK,IROT,ANGLE,
     &    YCLR,LAMBDA,ILOC,M)
C
C  THIS SUBROUTINE PLOTS THE MATCHING NETWORKS AT THE DIPOLE
C  FEEDPOINTS.
        REAL LANT,LAMBDA
        DIMENSION XO(20,20),YO(20,20),EL1(20,20)
        DIMENSION EL2(20,20),EL3(20,20),MTYPE(20,20)
C
C       WRITE(0,*)'IN FEEDPNT; NROW,NCOL: ',NROW,NCOL
C
        DO 1 M=1,NROW
        DO 1 N=1,NCOL
C
C       WRITE(0,*)'Executing FEEDPNT, MTYPE(M,N): ',MTYPE(M,N)
        IF(MTYPE(M,N).LT.9.OR.MTYPE(M,N).GT.16)GO TO 1
C
        MM=MTYPE(M,N)-8
        GOTO(9,10,11,12,13,14,15,16)MM
C
    9   CALL NET09(XO(M,N),YO(M,N),EL1(M,N),EL2(M,N),EL3(M,N),
     &    LANT,WANT,WFEED,THICK,EPSR,FTHICK,IROT,ANGLE,
     &    YCLR,LAMBDA,ILOC,M,NROW)
        GO TO 1
C
   10   CALL NET10(XO(M,N),YO(M,N),EL1(M,N),EL2(M,N),EL3(M,N),
     &    LANT,WANT,WFEED,THICK,EPSR,FTHICK,IROT,ANGLE,
     &    YCLR,LAMBDA,ILOC,M,NROW)
        GO TO 1
C
   11   CALL NET11(XO(M,N),YO(M,N),EL1(M,N),EL2(M,N),EL3(M,N),
     &    LANT,WANT,WFEED,THICK,EPSR,FTHICK,IROT,ANGLE,
     &    YCLR,LAMBDA,ILOC,M,NROW)
        GO TO 1
C
   12   CALL NET12(XO(M,N),YO(M,N),EL1(M,N),EL2(M,N),EL3(M,N),
     &    LANT,WANT,WFEED,THICK,EPSR,FTHICK,IROT,ANGLE,
     &    YCLR,LAMBDA,ILOC,M,NROW)
        GO TO 1
```

```
C
   13 CALL NET13(XO(M,N),YO(M,N),EL1(M,N),EL2(M,N),EL3(M,N),
     &   LANT,WANT,WFEED,THICK,EPSR,FTHICK,IROT,ANGLE,
     &   YCLR,LAMBDA,ILOC,M,NROW)
      GO TO 1
C
   14 CALL NET14(XO(M,N),YO(M,N),EL1(M,N),EL2(M,N),EL3(M,N),
     &   LANT,WANT,WFEED,THICK,EPSR,FTHICK,IROT,ANGLE,
     &   YCLR,LAMBDA,ILOC,M,NROW)
      GO TO 1
C
   15 CALL NET15(XO(M,N),YO(M,N),EL1(M,N),EL2(M,N),EL3(M,N),
     &   LANT,WANT,WFEED,THICK,EPSR,FTHICK,IROT,ANGLE,
     &   YCLR,LAMBDA,ILOC,M,NROW)
      GO TO 1
C
   16 CALL NET16(XO(M,N),YO(M,N),EL1(M,N),EL2(M,N),EL3(M,N),
     &   LANT,WANT,WFEED,THICK,EPSR,FTHICK,IROT,ANGLE,
     &   YCLR,LAMBDA,ILOC,M,NROW)
      GO TO 1
C
    1 CONTINUE
C     WRITE(0,*)'YCLR: ',YCLR
      RETURN
      END
C
C
C
      SUBROUTINE FEEDLIN(NROW,NCOL,XO,YO,WFEED,IROT,ANGLE,YCLR,
     &                   ITOP,YDIST)
C
C SUBROUTINE DRAWS FEEDLINE STRUCTURE
      DIMENSION XO(20,20),YO(20,20)
      DATA LTOP/1/,LBOT/2/
C
      DO 1 I=1,NROW
      DO 1 J=1,NCOL
C
C DRAW THE FEED-LINE EXTENSION
      IF(ITOP.NE.1)THEN
         IF(I.NE.NROW)THEN
            CALL BOX(XO(I,J),YO(I,J)-YCLR,2.*YCLR,WFEED,2,
     &               IROT,XO(I,J),YO(I,J),ANGLE,LBOT)
            CALL BOX(XO(I,J),YO(I,J)-YCLR,2.*YCLR,WFEED,2,
     &               IROT,XO(I,J),YO(I,J),ANGLE,LTOP)
         ELSE
            CALL BOX(XO(I,J),YO(I,J)-YCLR,YCLR,WFEED,2,
     &               IROT,XO(I,J),YO(I,J),ANGLE,LBOT)
            CALL BOX(XO(I,J),YO(I,J)-YCLR,YCLR,WFEED,2,
     &               IROT,XO(I,J),YO(I,J),ANGLE,LTOP)
         ENDIF
C
      ELSE
         IF(I.NE.1)THEN
            CALL BOX(XO(I,J),YO(I,J)-YCLR,2.*YCLR,WFEED,2,
     &               IROT,XO(I,J),YO(I,J),ANGLE,LBOT)
            CALL BOX(XO(I,J),YO(I,J)-YCLR,2.*YCLR,WFEED,2,
     &               IROT,XO(I,J),YO(I,J),ANGLE,LTOP)
         ELSE
            CALL BOX(XO(I,J),YO(I,J),YCLR,WFEED,2,
     &               IROT,XO(I,J),YO(I,J),ANGLE,LBOT)
            CALL BOX(XO(I,J),YO(I,J),YCLR,WFEED,2,
     &               IROT,XO(I,J),YO(I,J),ANGLE,LTOP)
         ENDIF
C
      ENDIF
C
    1 CONTINUE
C
```

```
C
      RETURN
      END

C
C
      SUBROUTINE BOX(XO,YO,LENG,WIDTH,ITYPE,IROT,RXO,RYO,ANGLE,LAYER)
C
C SUBROUTINE DRAWS A BOX AT ORIGIN XO,YO OF SIZE LENG BY WIDTH.
C IF IROT .NE. 0, THE BOX IS ROTATED ABOUT ORIGIN RXO,RYO.
C THE ORIANTATION OF THE BOX BEFORE ROTATION IS:
C    ITYPE=1, 0   DEGREES
C    ITYPE=2, 90  DEGREES
C    ITYPE=3, 180 DEGREES
C    ITYPE=4, 270 DEGREES
C
C
      REAL LENG
      DIMENSION X(5),Y(5),IP(5)
      COMMON /COORD/NP1,NP2,NP3,X1(12000),X2(12000),X3(1100),
     &                          Y1(12000),Y2(12000),Y3(1100),
     &                          IP1(12000),IP2(12000),IP3(1100)
C
C
      DATA LMIN/1/,LMAX/3/,EPS/1E-10/
C
      IF(ITYPE.LT.1.OR.ITYPE.GT.4)THEN
         ITYPE=1
      ENDIF
C
      IF(LAYER.LT.LMIN.OR.LAYER.GT.LMAX)THEN
         LAYER=LMIN
      ENDIF
C
      W2=WIDTH/2.
C
      GO TO (10,20,30,40)ITYPE
C
C 0 DEGREE ORIENTATION
C
   10 X(1)=XO
      Y(1)=YO-W2
      IP(1)=3
C
      X(2)=XO+LENG
      Y(2)=Y(1)
      IP(2)=2
C
      X(3)=X(2)
      Y(3)=Y(2)+WIDTH
      IP(3)=2
C
      X(4)=XO
      Y(4)=Y(3)
      IP(4)=2
C
      X(5)=X(1)
      Y(5)=Y(1)
      IP(5)=2
C
      GO TO 100
C
C 90 DEGREE ORIENTATION
C
   20 X(1)=XO+W2
      Y(1)=YO
      IP(1)=3
C
      X(2)=X(1)
```

```
      Y(2)=YO+LENG
      IP(2)=2
C
      X(3)=XO-W2
      Y(3)=Y(2)
      IP(3)=2
C
      X(4)=X(3)
      Y(4)=YO
      IP(4)=2
C
      X(5)=X(1)
      Y(5)=Y(1)
      IP(5)=2
C
      GO TO 100
C
C  180 DEGREE ORIENTATION
C
   30 X(1)=XO
      Y(1)=YO+W2
      IP(1)=3
C
      X(2)=XO-LENG
      Y(2)=Y(1)
      IP(2)=2
C
      X(3)=X(2)
      Y(3)=Y(2)-WIDTH
      IP(3)=2
C
      X(4)=XO
      Y(4)=Y(3)
      IP(4)=2
C
      X(5)=X(1)
      Y(5)=Y(1)
      IP(5)=2
C
      GO TO 100
C
C
C  270 DEGREE ORIENTATION
C
   40 X(1)=XO-W2
      Y(1)=YO
      IP(1)=3
C
      X(2)=X(1)
      Y(2)=YO-LENG
      IP(2)=2
C
      X(3)=XO+W2
      Y(3)=Y(2)
      IP(3)=2
C
      X(4)=X(3)
      Y(4)=YO
      IP(4)=2
C
      X(5)=X(1)
      Y(5)=Y(1)
      IP(5)=2
C
C
  100 CONTINUE
C
C
C  THIS SECTION OF CODE ROTATES THE COORDINATES OF THE BOX ABOUT
```

```
C  AN ARBITRARY POINT RXO, RYO.  THE VARIABLE IROT IS SET TO ONE
C  TO INDICATE ROTATION.  THE ANGLE OF ROTATION IS IN RADIANS MEASURED
C  IN THE NORMAL MATHEMATICAL SENSE.
C
      IF(IROT.EQ.1)THEN
         DO 2 J=1,5
         XH=X(J)
         YH=Y(J)
         RV=SQRT((X(J)-RXO)**2+(Y(J)-RYO)**2)
         THETA=ATAN2(Y(J)-RYO,X(J)-RXO+EPS)
         TNEW=THETA+ANGLE
         X(J)=RV*COS(TNEW)+RXO
    2    Y(J)=RV*SIN(TNEW)+RYO
C        WRITE(0,*)'NON-ROTATED COORDINATES: ',XH,YH
C   2    WRITE(0,*)'PRINTING ROTATED COORDINATES.....',X(J),Y(J)
      ENDIF
C
C
C
C  NOW THE DATA (ROTATED IF DESIRED) ARE WRITTEN INTO THREE
C  ARRAYS CORRESPONDING TO THE THREE LAYERS: TOP LAYER, BOTTON
C  LAYER, AND VIA LAYER.  PEN LIFT INFORMATION IS ALSO RECORDED
C  IN AN ARRAY FOR EACH LAYER.
C
      DO 1 J=1,5
C
      GOTO(50,60,70)LAYER
C
   50 X1(NP1)=X(J)
      Y1(NP1)=Y(J)
      IP1(NP1)=IP(J)
      NP1=NP1+1
      GO TO 1
C
   60 X2(NP2)=X(J)
      Y2(NP2)=Y(J)
      IP2(NP2)=IP(J)
      NP2=NP2+1
      GO TO 1
C
   70 X3(NP3)=X(J)
      Y3(NP3)=Y(J)
      IP3(NP3)=IP(J)
      NP3=NP3+1
C
    1 CONTINUE

C     II=II+1
C     IF(II/100*100.EQ.II)THEN
C        WRITE(0,*)'NP1,NP2,NP3 ',NP1,NP2,NP3
C     ENDIF
C
      RETURN
      END
C
C
      SUBROUTINE DRAWANT
C
C  THIS SUBROUTINE PLOTS THE COORDINATES STORED IN THE
C  COORDINATE ARRAYS X1,X2,X3,Y1,Y2, AND Y3 UNDER CONTROL
C  OF THE PEN-LIFT PARAMETERS IN IP1,IP2, AND IP3.
C
      COMMON /COORD/NP1,NP2,NP3,X1(12000),X2(12000),X3(1100),
     &                          Y1(12000),Y2(12000),Y3(1100),
     &                          IP1(12000),IP2(12000),IP3(1100)
      COMMON /WINSIZ/WX1,WY1,WX2,WY2,FACT
C
      DATA XMAX/-1E10/,XMIN/1E10/,YMAX/-1E10/,YMIN/1E10/
C
```

```
      WRITE(0,100)
  100 FORMAT(1H ,'Plotting.... '/)
C
C
      CALL COLOR(1,IERR)
      DO 1 J=1,NP1-1
         IF(X1(J).GE.XMAX)XMAX=X1(J)
         IF(X1(J).LE.XMIN)XMIN=X1(J)
         IF(Y1(J).GE.YMAX)YMAX=Y1(J)
         IF(Y1(J).LE.YMIN)YMIN=Y1(J)
         CALL PLOT(X1(J),Y1(J),IP1(J))
    1 CONTINUE
C
      CALL PLOT(0.0,0.0,-999)
C     WRITE(0,*)'WX1,WY1,WX2,WY2: ',WX1,WY1,WX2,WY2
C
      BX1=WX1
      BX2=WX2
      BY1=WY1
      BY2=WY2
C
      CALL WINDOW(BX1,BY1,BX2,BY2)
      CALL FACTOR(FACT)
C
      CALL COLOR(2,IERR)
C
      DO 2 J=1,NP2-1
         CALL PLOT(X2(J),Y2(J),IP2(J))
    2 CONTINUE
C
      CALL PLOT(0.0,0.0,-999)
C
      BX1=WX1
      BX2=WX2
      BY1=WY1
      BY2=WY2
C
C     WRITE(0,*)'WX1,WY1,WX2,WY2: ',WX1,WY1,WX2,WY2
      CALL WINDOW(BX1,BY1,BX2,BY2)
      CALL FACTOR(FACT)
C
      IF(NP3.GT.1)THEN
         CALL COLOR(3,IERR)
         DO 3 J=1,NP3-1
            CALL PLOT(X3(J),Y3(J),IP3(J))
    3    CONTINUE
      ENDIF
C
  777 CALL PLOT(0.,0.,999)
C
      WRITE(0,110)NP1-1,NP2-1,NP3-1
  110 FORMAT(1H ,'Coordinates Plotted: '/1H ,
     &    6X,'Top Layer:    ',I10/1H ,
     &    6X,'Bottom Layer: ',I10/1H ,
     &    6X,'Cross-Over:   ',I10/)
C
      WRITE(0,200)XMIN,XMAX,YMIN,YMAX
  200 FORMAT(1H ,6X,'Minimum X-Coordinate: ',F12.3/1H,
     &           6X,'Maximum X-Coordinate: ',F12.3/1H,
     &           6X,'Minimum Y-Coordinate: ',F12.3/1H,
     &           6X,'Maximum Y-Coordinate: ',F12.3/)
C
      RETURN
      END
C
C
      BLOCK DATA INITIL
C  BLOCK DATA SUBPROGRAM TO INITILIZE COMMON BLOCK
      COMMON /COORD/NP1,NP2,NP3,X1(12000),X2(12000),X3(1100),
```

```
     &                          Y1(12000),Y2(12000),Y3(1100),
     &                          IP1(12000),IP2(12000),IP3(1100)
      DATA NP1/1/,NP2/1/NP3/1/
      END
C
C
      SUBROUTINE MEANDER(X,Y,XP1,YP1,NPOINT,YCLR,YSPAC,ANGLE,
     &                   WNORM,LAMBDA,EPSR,RFLAG)
C
      LOGICAL RFLAG
      DIMENSION X(30),Y(30),XM(30),YM(30)
      REAL LD,LG,LAMBDA
      DATA PIOVR2/1.570796327/,EPS/1E-15/
C
C  THIS SUBPROGRAM DETERMINES THE DIMENSIONS OF A MEANDERING
C  TRANSMISSION LINE OF THE LENGTH NEEDED TO MAKE THE SPACING
C  BETWEEN ELEMENTS TO BE A MULTIPLE OF ONE-HALF WAVELENGTH.
C
C  YSPAC IS THE DISTANCE BETWEEN ANTENNA CENTERS.
C  YCLR IS THE FEED-LINE EXTENSION AT THE ANTENNA.
C  DDIR IS THE DIRECT DISTANCE BETWEEN EXTENSIONS (ONE ANTENNA
C  TO THE NEXT).
C  DTOT IS THE TOTAL DIRECT PATH DISTANCE INCLUDING THE EXTENSIONS.
C
C
      XP1=-YCLR*SIN(ANGLE)
      YP1= YCLR*COS(ANGLE)
      XP2=-XP1
      YP2= YSPAC-YP1
      DDIR=SQRT((XP2-XP1)**2+(YP2-YP1)**2)
      DTOT=DDIR+2.*YCLR
C
C     WRITE(0,*)'YSPAC: ',YSPAC
C     WRITE(0,*)'XP1,XP2,YP1,YP2: ',XP1,XP2,YP1,YP2
C     WRITE(0,*)'DTOT: ',DTOT
C
C  COMPARE THIS DISTANCE WITH THAT NEEDED TO MAKE INTEGRAL MULTIPLES
C  OF ONE-HALF WAVE.
C
C  CHECK THE SPECIAL CASE--IS THE LENGTH ALREADY WITHIN 1/100 WAVELENGTH
C  OF THE CORRECT DISTANCE.
C
C  LG IS THE WAVELENGTH ALONG THE TRANSMISSION LINE
      LG=LAMBDA/SQRT(EPSR)
C  WLIM IS THE MAXIMUM ALLOWABLE WIDTH OF THE MEANDER
      WLIM=WNORM*LG
C
C     DO 1 N=1,10
C        IF(ABS(DTOT-FLOAT(N)*.5*LG).LE.0.01*LG)THEN
C           GO TO 100
C        ENDIF
C   1 CONTINUE
C
      DO 2 N=1,10
         LD=FLOAT(N)*.5*LG
C        WRITE(0,*)'N, DTOT, LG, LD: ',N,DTOT,LG,LD
         IF(LD.GE.DTOT)THEN
           GO TO 200
         ENDIF
    2 CONTINUE
C
C     STOP 'UNABLE TO EXTEND FEED LINE TO MULTIPLE OF HALF-WAVE.'
C
C  CALCULATE PARAMETERS OF THE MEANDER LINE.
  200 CONTINUE
C        WRITE(0,*)'N,DTOT,LG,LD,YCLR: ',N,DTOT,LG,LD,YCLR
C  COMPUTE THE LENGTH OF THE MEANDERED SECTION OF THE LINE
      DMEA=LD-2.*YCLR
C     WRITE(0,*)'MEANDERED LNG: ',DMEA
```

```
C     WRITE(0,*)'DIRECT COURSE DISTANCE: ',DDIR
C
C  THE DIRECT COURSE DISTANCE IS DDIR
C
      DO 3 N=4,20,4
         W=2.*SQRT(DMEA**2-DDIR**2)/FLOAT(N)
C        WRITE(0,*)'DMEA, DDIR, WLIM, W: ',DMEA,DDIR,WLIM,W
         IF(W.LE.WLIM)THEN
            NSEGS=N
C           WRITE(0,*)'W,NSEGS: ',W,NSEGS
            GO TO 300
         ENDIF
    3 CONTINUE
C
      WRITE(0,*)'*** ERROR:  Meander Line Exceeds 20 Segments.'
C
C  THIS CODE CREATES THE MEANDER LINE.
  300 CONTINUE
C     WRITE(0,*)'SEGMENTS: ',NSEGS
C
C
      W2=W/2.
      DELTAX=DDIR/FLOAT(NSEGS)
C
C     WRITE(0,*)'DELTAX, W2: ',DELTAX,W2
      DO 4 N=1,NSEGS+1
C
C  CODE TO MAKE TAKE-OFF FROM MATCHING NETWORK NOT PRALLEL TO ANTENNA.
         IF(ANGLE.LT.0.)THEN
            XM(N)=+W2*SIN(PIOVR2*FLOAT(N-1))
         ELSE
            XM(N)=-W2*SIN(PIOVR2*FLOAT(N-1))
         ENDIF
C
         YM(N)=DELTAX*(FLOAT(N-1))
C
C        WRITE(0,111)XM(N),YM(N)
C 111    FORMAT(1H ,'COORDINATES; ',2F10.3)
    4 CONTINUE
C
C     CALL PLOT(0.0,0.0,999)
C  ROTATE THE MEANDER LINE AT THE BOTTON END BY THE ANGLE
C  NECESSARY TO CONNECT TO THE NEXT FEED NETWORK.
C
      IF(ABS(XP2-XP1).LE.1E-8)THEN
         TH1=0.
      ELSE
         TH1=ATAN2(YP2-YP1,XP2-XP1)-PIOVR2
      ENDIF
C
C     WRITE(0,*)'ANGLE TH1: ',TH1
C
C     WRITE(0,444)
C 444 FORMAT(1H ,'ENTER ANGLE OF ROTATION: '\)
C     READ(0,*)TH1
C
      IF(RFLAG)TH1=-PIOVR2
C
      DO 5 N=1,NSEGS+1
         RV=SQRT((XM(N)-XM(1))**2+(YM(N)-YM(1))**2)
         TH2=ATAN2(YM(N)-YM(1)+EPS,XM(N)-XM(1)+EPS)
         TNEW=TH2+TH1
         X(N)=RV*COS(TNEW)+XM(1)
         Y(N)=RV*SIN(TNEW)+YM(1)
    5 CONTINUE
C
C  ZERO WIDTH MEANDER LINE HAS BEEN CREATED AND ROTATED.
C  A CALL TO THE SUBROUTINE PATH WILL CHANGE THIS ZERO
C  WIDTH LINE INTO A SERIES OF CONNECTED BOXES.
```

```
C
      NPOINT=NSEGS+1
C
      RETURN
      END
C
C
      SUBROUTINE TEE(XO,YO,XP1,YP1,NROW,NCOL,YDIST,
     &               YCLR,WIDTH,LAYER,ITOP)
C
      DIMENSION XO(20,20),YO(20,20)
C
C     WRITE(0,*)'IN SUBROUTINE TEE...'
C  DRAW THE T-CONNECTORS
      DO 2 N=1,NCOL
C
      IF(ITOP.EQ.1)THEN
         XF=XO(NROW,N)+XP1
         YF=YO(NROW,N)+YDIST
C
         IF(N.NE.NCOL)THEN
            CALL BOX(XF,YF,YCLR,WIDTH,1,0,XF,YF,
     &               0.,LAYER)
         ELSE
            CALL BOX(XF,YF,YCLR,WIDTH,1,0,XF,YF,
     &               0.,LAYER)
         ENDIF
C
C        CALL BOX(XF,YF,YCLR,WIDTH,4,0,XF,YF,
C    &            0.,LAYER)
C
         IF(N.NE.1)THEN
            CALL BOX(XF,YF,YCLR,WIDTH,3,0,XF,YF,
     &               0.,LAYER)
C
         ENDIF
C
      ELSE
         XF=XO(1,N)-XP1
         YF=YO(1,N)-YDIST
C
         IF(N.NE.NCOL)THEN
            CALL BOX(XF,YF,YCLR,WIDTH,1,0,XF,YF,
     &               0.,LAYER)
         ELSE
C           CALL BOX(XF,YF,YCLR,WIDTH,1,0,XF,YF,
C    &               0.,LAYER)
         ENDIF
C
         CALL BOX(XF,YF,YCLR,WIDTH,2,0,XF,YF,
     &            0.,LAYER)
C
         IF(N.NE.1)THEN
            CALL BOX(XF,YF,YCLR,WIDTH,3,0,XF,YF,
     &               0.,LAYER)
         ENDIF
C
      ENDIF
C
      IF(ITOP.NE.1)THEN
         IF(N.EQ.NCOL/2+1 )THEN
            CALL BOX(XF,YF,YCLR,WIDTH,4,0,XF,YF,0.,LAYER)
         ENDIF
      ELSE
         IF(N.EQ.NCOL/2+1)THEN
            CALL BOX(XF,YF,YCLR,WIDTH,2,0,XF,YF,0.,LAYER)
         ENDIF
      ENDIF
C
```

```
C
    2 CONTINUE
C
      RETURN
      END
C
C
      SUBROUTINE PATH(XO,YO,NROW,NCOL,XP,YP,XP1,YP1,NPOINT,WIDTH,LAYER)
C
C SUBROUTINE DRAWS AN ANGLED BOX BETWEEN POINTS IN ARRAY XP,YP.
C REVISION OF 9/8/89
C
      COMMON /COORD/NP1,NP2,NP3,X1(12000),X2(12000),X3(1100),
     &                          Y1(12000),Y2(12000),Y3(1100),
     &                          IP1(12000),IP2(12000),IP3(1100)
C
      DIMENSION XP(NPOINT),YP(NPOINT)
      DIMENSION XO(20,20),YO(20,20)
C
      DATA LMIN/1/,LMAX/3/
      DATA PI/3.141592654/,MAXSEG/8/
C
C     WRITE(0,*)'IN PATH, WIDTH, LAYER: ',WIDTH,LAYER
      IF(LAYER.LT.LMIN.OR.LAYER.GT.LMAX)THEN
         LAYER=LMIN
      ENDIF
C
C     WRITE(0,*)'NPOINT: ',NPOINT
C     WRITE(0,*)'NROW,NCOL: ',NROW,NCOL
      W2=WIDTH/2.
C
      DO 2 I=1,NROW-1
      DO 2 J=1,NCOL
C
      DO 1 N=1,NPOINT-1
C     WRITE(0,*)'NP1,NP2,NP3: ',NP1,NP2,NP3
C
      DX=XP(N+1)-XP(N)
      DY=YP(N+1)-YP(N)
C
      ANGLE=ATAN2(DY,DX)
C
      COSA=W2*COS(ANGLE)
      SINA=W2*SIN(ANGLE)
C
      IF(LAYER.LT.1.OR.LAYER.GT.3)THEN
         WRITE(0,*)'LAYER: ',LAYER,' IS OUT OF RANGE.'
         LAYER=1
      ENDIF
C
C  WRITE FEEDLINE MEANDER DATA INTO MEMORY
C
      GO TO (10,20,30)LAYER
C
C
   10 X1(NP1)=XO(I,J)+XP1+XP(N)+SINA
      Y1(NP1)=YO(I,J)+YP1+YP(N)-COSA
      IP1(NP1)=3
      NP1=NP1+1
C
      X1(NP1)=XO(I,J)+XP1+XP(N+1)+SINA
      Y1(NP1)=YO(I,J)+YP1+YP(N+1)-COSA
      IP1(NP1)=2
      NP1=NP1+1
C
      DO 3 K=1,MAXSEG
         PHI=ANGLE+FLOAT(K)/FLOAT(MAXSEG)*PI
         X1(NP1)=XO(I,J)+XP1+XP(N+1)+W2*SIN(PHI)
         Y1(NP1)=YO(I,J)+YP1+YP(N+1)-W2*COS(PHI)
```

```
            IP1(NP1)=2
            NP1=NP1+1
    3 CONTINUE
C
         X1(NP1)=XO(I,J)+XP1+XP(N)-SINA
         Y1(NP1)=YO(I,J)+YP1+YP(N)+COSA
         IP1(NP1)=2
         NP1=NP1+1
C
         DO 4 K=1,MAXSEG
            PHI=ANGLE+FLOAT(K)/FLOAT(MAXSEG)*PI+PI
            X1(NP1)=XO(I,J)+XP1+XP(N)+W2*SIN(PHI)
            Y1(NP1)=YO(I,J)+YP1+YP(N)-W2*COS(PHI)
            IP1(NP1)=2
            NP1=NP1+1
    4 CONTINUE
C
         GO TO 1
C
C
   20 X2(NP2)=XO(I,J)+XP1+XP(N)+SINA
         Y2(NP2)=YO(I,J)+YP1+YP(N)-COSA
         IP2(NP2)=3
         NP2=NP2+1
C
         X2(NP2)=XO(I,J)+XP1+XP(N+1)+SINA
         Y2(NP2)=YO(I,J)+YP1+YP(N+1)-COSA
         IP2(NP2)=2
         NP2=NP2+1
C
         DO 5 K=1,MAXSEG
            PHI=ANGLE+FLOAT(K)/FLOAT(MAXSEG)*PI
            X2(NP2)=XO(I,J)+XP1+XP(N+1)+W2*SIN(PHI)
            Y2(NP2)=YO(I,J)+YP1+YP(N+1)-W2*COS(PHI)
            IP2(NP2)=2
            NP2=NP2+1
    5 CONTINUE
C
         X2(NP2)=XO(I,J)+XP1+XP(N)-SINA
         Y2(NP2)=YO(I,J)+YP1+YP(N)+COSA
         IP2(NP2)=2
         NP2=NP2+1
C
         DO 6 K=1,MAXSEG
            PHI=ANGLE+FLOAT(K)/FLOAT(MAXSEG)*PI+PI
            X2(NP2)=XO(I,J)+XP1+XP(N)+W2*SIN(PHI)
            Y2(NP2)=YO(I,J)+YP1+YP(N)-W2*COS(PHI)
            IP2(NP2)=2
            NP2=NP2+1
    6 CONTINUE
C
         GO TO 1
C
C
   30 X3(NP3)=XO(I,J)+XP1+XP(N)+SINA
         Y3(NP3)=YO(I,J)+YP1+YP(N)-COSA
         IP3(NP3)=3
         NP3=NP3+1
C
         X3(NP3)=XO(I,J)+XP1+XP(N+1)+SINA
         Y3(NP3)=YO(I,J)+YP1+YP(N+1)-COSA
         IP3(NP3)=2
         NP3=NP3+1
C
         DO 7 K=1,MAXSEG
            PHI=ANGLE+FLOAT(K)/FLOAT(MAXSEG)*PI
            X3(NP3)=XO(I,J)+XP1+XP(N+1)+W2*SIN(PHI)
            Y3(NP3)=YO(I,J)+YP1+YP(N+1)-W2*COS(PHI)
            IP3(NP3)=2
```

```
            NP3=NP3+1
    7 CONTINUE
C
      X3(NP3)=XO(I,J)+XP1+XP(N)-SINA
      Y3(NP3)=YO(I,J)+YP1+YP(N)+COSA
      IP3(NP3)=2
      NP3=NP3+1
C
      DO 8 K=1,MAXSEG
         PHI=ANGLE+FLOAT(K)/FLOAT(MAXSEG)*PI+PI
         X3(NP3)=XO(I,J)+XP1+XP(N)+W2*SIN(PHI)
         Y3(NP3)=YO(I,J)+YP1+YP(N)-W2*COS(PHI)
         IP3(NP3)=2
         NP3=NP3+1
    8 CONTINUE
C
C
    1 CONTINUE
    2 CONTINUE
C
C
      RETURN
C
      END
C
```

```
C
C ************************************************************************** C
C
C                   ELEMENT CALCULATION SUBROUTINES
C
C ************************************************************************** C
C
      SUBROUTINE SHUNTC(XORG,YORG,C,WCAP,WFEED,WMIN,
     &                THICK,EPSR,IROT,ANGLE,CLR,ITYPE,YCLR)
C
C  THIS SUBROUTINE DRAWS A SHUNT CAPACITOR BETWEEN THE
C  ANTENNA ELEMENTS OR ACROSS THE FEEDLINE AS SELECTED
C  BY ITYPE.
C       ITYPE = 1   A CAPACITOR IS DRAWN ACROSS THE FEEDLINE.
C       ITYPE = 2   A CAPACITOR IS DRAWN BETWEEN THE ANTENNAS.
C
      REAL L1,LSC,LSC1,LANT,LSC3,LCROS
      DATA OL/.05/,LTOP/1/,LBOT/2/,LCRS/3/
C
      IF(ITYPE.NE.1.AND.ITYPE.NE.2)RETURN
C
      IF(ITYPE.EQ.1)THEN
         CALL CAPVAL(WMIN,C,WSC,LSC,EPSR,THICK)
C
         WSC2=WSC/2.
         WF2=WFEED/2.
         WC2=WCAP/2.
C
C CAPACITOR PLATE CONNECTED TO TOP FEED LINE.
         X1=XORG-WSC2-CLR-WF2
         Y1=YORG+WC2-OL
         LSC1=LSC+OL+CLR
         CALL BOX(X1,Y1,LSC1,WSC,2,IROT,XORG,YORG,ANGLE,LTOP)
C
C CAPACITOR PLATE ON RIGHT SIDE CONNECTED BY CROSS-OVER
         X2=XORG+WSC2+CLR+WF2
         Y2=YORG+WC2+CLR
         CALL BOX(X2,Y2,LSC,WSC,2,IROT,XORG,YORG,ANGLE,LTOP)
C
C CAPACITOR PLATE CONNECTED TO BOTTOM FEED LINE.
         X3=XORG+WSC2+CLR+WF2
         Y3=Y1
         LSC3=LSC1
         CALL BOX(X3,Y3,LSC3,WSC,2,IROT,XORG,YORG,ANGLE,LBOT)
C
C CROSS-OVER ON LAYER 3 CONNECTING LEFT AND RIGHT TOP PLATES.
C
         X4=XORG-WF2-CLR-WSC2
         Y4=YORG+WC2+CLR+LSC/2.
         LCROS=WFEED+2.*CLR+WSC
         CALL BOX(X4,Y4,LCROS,LSC,1,IROT,XORG,YORG,ANGLE,LCRS)
C
      ENDIF
C
      IF(ITYPE.EQ.2)THEN
         CALL CAPVAL(WMIN,C,WSC,LSC,EPSR,THICK)
C
         WSC2=WSC/2.
         WF2=WFEED/2.
         WC2=WCAP/2.
C
C CAPACITOR PLATE CONNECTED TO LEFT-HAND (BOTTOM) ANTENNA.
         X1=XORG-WSC2-CLR-WF2
         Y1=YORG-WC2+OL
         LSC1=LSC+OL+CLR
         CALL BOX(X1,Y1,LSC1,WSC,4,IROT,XORG,YORG,ANGLE,LBOT)
C
C CAPACITOR PLATE CONNECTED TO RIGHT-HAND (TOP) ANTENNA.
         X3=XORG+WSC2+CLR+WF2
```

```
         Y3=Y1
         LSC3=LSC1
         CALL BOX(X3,Y3,LSC3,WSC,4,IROT,XORG,YORG,ANGLE,LTOP)
C
C  CAPACITOR PLATE ON LEFT SIDE CONNECTED BY CROSS-OVER
         X2=X1
         Y2=YORG-WC2-CLR
         CALL BOX(X2,Y2,LSC,WSC,4,IROT,XORG,YORG,ANGLE,LTOP)
C
C  CROSS-OVER ON LAYER 3 CONNECTING LEFT AND RIGHT TOP PLATES.
C
         X4=XORG-WF2-CLR-WSC2
         Y4=YORG-WC2-CLR-LSC/2.
         LCROS=WFEED+2.*CLR+WSC
         CALL BOX(X4,Y4,LCROS,LSC,1,IROT,XORG,YORG,ANGLE,LCRS)
C
      ENDIF
C
C  DETERMINE LENGTH OF TLINE TO CLEAR ALL MATCHING ELEMENTS
C
      YCL=WC2+CLR+LSC+WFEED
C
      IF(YCL.GE.YCLR)YCLR=YCL
C
      RETURN
C
      END
C
C
      SUBROUTINE CAPVAL(WANT,CC,WC,LC,EPSR,THICK)
C  THIS SUBROUTINE CALCULATES THE DIMENSIONS OF A METAL FOIL
C  CAPACITOR FROM GIVEN MILAR PROPERTIES.
      REAL LC
C
      DIMENSION WIDTHS(6)
      DATA WIDTHS/.125,.25,.375,.5,.75,1.0/
C
C     WRITE(0,*)'EPSR, C (pF): ',EPSR,CC
      C=CC*1E-12
      TH=THICK*2.54E-5
      A=C*TH/(8.854E-12*EPSR)
      A=A/(.0254**2)
C
      X=SQRT(A)
C     WRITE(0,*)'A,X ',A,X
      IF(X.LE.WANT)THEN
         WC=WANT
         LC=A/WC
         RETURN
      ENDIF
C
      DO 1 J=1,6
      IF(X.LE.WIDTHS(J))THEN
         WC=WIDTHS(J)
         LC=A/WC
C        WRITE(0,*)'WIDTH/LENGTH OF LINE: ',WC,LC
         RETURN
      ENDIF
    1 CONTINUE
C
C  USE LARGEST WIDTH MATERIAL
      WC=WIDTHS(6)
      LC=A/WC
      RETURN
      END
C
C
      SUBROUTINE XNDVAL(L,O,D,LAMBDA,W,T)
C
```

```
C  CALLING FORM IN VARIOUS PARTS OF THESE PROGRAMS
C
C     SUBROUTINE INDVAL(L,W,D,LAMBDA,WIND,FTHICK)
C
C  L IS THE INDUCTANCE IN nH.
C  O IS THE OUTSIDE DIMENSION OF THE HORSE SHOE.
C  D IS THE DEPTH OF THE HORSE SHOE.
C  W IS THE WIDTH OF THE STRIP.
C  T IS THE THICKNESS OF THE METAL FOIL.
C  LAMBDA IS THE WAVELENGTH AT THE FREQUENCY OF OPERATION.
C
      REAL L,LAMBDA
      DATA DELTAD/0.0005/,ERROR/0.001/,DMIN/0.01/
C
      NTRY=0
C GUESS A VALUE
      D1=L/20.
C
   10 NTRY=NTRY+1
      U1=UIND(O,W,D1,T)
      Y1=L-U1
      U2=UIND(O,W,D1+DELTAD,T)
      Y2=L-U2
      IF(U1.LT.0..OR.U2.LT.0.)THEN
         D=DMIN
C        WRITE(0,100)
  100    FORMAT(1H ,'*** WARNING - Inductance Too Small'/1H ,
     &          '                  0.01 Inch U-Depth Used')
C
         RETURN
      ELSE
         D2=D1-DELTAD*Y1/(Y2-Y1)
      ENDIF
C     WRITE(0,*)'D1,D2: ',D1,D2
C
      IF(ABS(D2-D1).LE.ERROR)THEN
         D=D2
         IF(D.LT.DMIN)THEN
            D=DMIN
C           WRITE(0,100)
         ENDIF
C
         IF(D.GT.0.15*LAMBDA)THEN
            D=0.15*LAMBDA
C           WRITE(0,200)
  200       FORMAT(1H ,'*** WARNING - Inductance Too Large'/1H ,
     &             '                  0.15 Lambda Long Lines Used')
         ENDIF
C
         RETURN
      ELSE
         D1=D2
         GO TO 10
      ENDIF
C
C
      RETURN
      END
C
C
      REAL FUNCTION UIND(O,W,D,T)
C
C  CALCULATION OF INDUCTANCE OF U-SHAPED COILS
      REAL L11,L12,L22,L33,IND
C
C     O      Is the outer dimensions of the U-shape.
C     W      Is the width of the flat strip conductor.
C     D      Is the depth of the U-shaped conductor. (inside the U)
C     T      Is the thickness of the flat conductor.
```

```
C
C     WRITE(0,100)
C 100 FORMAT(1H ,'Enter O,D,W,T:  '\)
C     READ(0,*)O,D,W,T
C
      X1=D+W
C
C  CHECK TO SEE IF THE INDUCTANCE IS TOO SMALL TO BE REALIZED.
C
      IF(X1.LE.0.)THEN
         UIND=-1.
         RETURN
      ENDIF
C
      X3=X1
      X2=O-2*W
C
      L11=5.08*X1*(LOG(2.*X1/(W+T)) + (W+T)/(3.*X1) + .50049)
C
      L22=5.08*X2*(LOG(2.*X2/(W+T)) + (W+T)/(3.*X2) + .50049)
C
      L33=5.08*X3*(LOG(2.*X3/(W+T)) + (W+T)/(3.*X3) + .50049)
C
      X=(O-W)/W
      GMDL=ALOG(X*W)-(1./(12.*X**2) + 1./(60.*X**4) + 1./(168.*X**6)
     &   + 1./(360.*X**8) + 1./(660.*X**10) )
      GMD=EXP(GMDL)
C
      Y=X1/GMD
      Q=ALOG(Y + SQRT(1.+Y**2)) - SQRT(1.+1./Y**2) + 1./Y
C
      L12=5.08*X1*Q
C
      UIND=L11+L22+L33-2.*L12
C
C     WRITE(0,200)IND
C 200 FORMAT(1H ,'Inductance (nH): ',1F8.1)
C
      RETURN
      END
C
      SUBROUTINE INDVAL(L,O,D,LAMBDA,W,T)
C
C  L IS THE INDUCTANCE IN nH.
C  W IS THE OUTSIDE DIMENSION OF THE HORSE SHOE.
C  D IS THE DEPTH OF THE HORSE SHOE.
      REAL L,LAMBDA
C     WRITE(0,*)'L,W: ',L,W
C
C  DETERMINE SLOPE AND INTERCEPT OF INDUCTANCE EQUATIONS
      XINT=-2.655+13.861*W
      SLOP= 7.659+40.38*W
C  CALCULATE DEPTH OF HORSE SHOE
      D=(L-XINT)/SLOP
C     WRITE(0,*)'L, D ',L,D
      IF(D.LE..025)D=.025
      IF(D.GT.0.15*LAMBDA)D=0.15*LAMBDA
C
      RETURN
      END
C
      SUBROUTINE SHUNTL(XORG,YORG,WFEED,WIND,FTHICK,WC,LC,
     &                  CLR,L,LAMBDA,ITYPE,IROT,ANGLE,YCLR)
C
C   THIS SUBROUTINE DRAWS A SHUNT INDUCTOR BETWEEN THE
C   ANTENNA ELEMENTS OR ACROSS THE FEEDLINE AS SELECTED
C   BY ITYPE.
C        ITYPE = 1   AN INDUCTOR IS DRAWN ACROSS THE FEEDLINE.
C        ITYPE = 2   AN INDUCTOR IS DRAWN BETWEEN THE ANTENNAS.
```

```
C
      REAL LC,LC2,L,LAMBDA
      DATA LTOP/1/,LBOT/2/,LBRD/3/
C
C DRAW THE INDUCTORS
C
      IF(ITYPE.NE.1.AND.ITYPE.NE.2.)RETURN
C
      LC2=LC/2.
      WF2=WFEED/2.
      WIND2=WIND/2.
      WIND4=WIND/4.
      WC2=WC/2.
C
      IF(ITYPE.EQ.2)THEN
C
C DRAW THE ANTENNA INDUCTOR (BOTTOM)
C
C GET THE PARAMETERS OF THE INDUCTOR
         IF(L.LE.8.)THEN
            X1=XORG-CLR-WF2-WIND2
            X2=XORG+CLR+WF2+WIND2
            CALL INDVAL(L,2.*(WF2+CLR)+WIND,D,LAMBDA,WIND,FTHICK)
         ELSEIF(L.GT.8..AND.L.LE.20.)THEN
            X1=XORG-WF2-CLR-LC2
            X2=XORG+WF2+CLR+LC2
            CALL INDVAL(L,X2-X1+WIND,D,LAMBDA,WIND,FTHICK)
         ELSE
            X1=XORG-WF2-CLR-LC2
            X2=XORG+WF2+CLR+LC2
            CALL INDVAL(L,X2-X1+WIND,D,LAMBDA,WIND,FTHICK)
         ENDIF
C
         Y1=YORG-WC2+WIND4
         Y2=Y1
         X3=X1-WIND2
         Y3=YORG-WC2-D-WIND2
C
         CALL BOX(X1,Y1,D+WIND+WIND4,WIND,4,IROT,XORG,YORG,ANGLE,LBOT)
         CALL BOX(X2,Y1,D+WIND+WIND4,WIND,4,IROT,XORG,YORG,ANGLE,LTOP)
         CALL BOX(X3,Y3,X2-X1+WIND,WIND,1,IROT,XORG,YORG,ANGLE,LBRD)
C
C MAKE A VIA
C        CALL VIA(X1+WIND2,Y1,IROT,ANGLE,LVIA)
C
      ENDIF
C
      IF(ITYPE.EQ.1)THEN
C
C DRAW THE FEEDLINE INDUCTOR (TOP)
C
C GET THE PARAMETERS OF THE INDUCTOR
         IF(L.LE.8.)THEN
            X1=XORG+WF2+CLR+WIND2
            X2=XORG-WF2-CLR-WIND2
            CALL INDVAL(L,2.*(WF2+CLR)+WIND,D,LAMBDA,WIND,FTHICK)
         ELSEIF(L.GT.8..AND.L.LE.20.)THEN
            X1=XORG+WF2+CLR+LC2
            X2=XORG-WF2-CLR-LC2
            CALL INDVAL(L,X1-X2+WIND,D,LAMBDA,WIND,FTHICK)
         ELSE
            X1=XORG+WF2+CLR+LC2
            X2=XORG-WF2-CLR-LC2
            CALL INDVAL(L,X1-X2+WIND,D,LAMBDA,WIND,FTHICK)
         ENDIF
C
         Y1=YORG+WC2-WIND4
         Y2=Y1
         X3=X1+WIND2
```

```
          Y3=YORG+WC2+D+WIND2
C
C
         CALL BOX(X1,Y1,D+WIND+WIND4,WIND,2,IROT,XORG,YORG,ANGLE,LBOT)
         CALL BOX(X2,Y1,D+WIND+WIND4,WIND,2,IROT,XORG,YORG,ANGLE,LTOP)
         CALL BOX(X3,Y3,X1-X2+WIND,WIND,3,IROT,XORG,YORG,ANGLE,LBRD)
C
C MAKE A VIA
C      CALL VIA(X1-WIND2,Y1,IROT,ANGLE,LVIA)
C
      ENDIF
C
C DETERMINE LENGTH OF TLINE TO CLEAR ALL MATCHING ELEMENTS
      YCL=AMAX1(WC2,YORG-Y3+WIND2,Y3-YORG+WIND2)+WFEED
C      YCL=AMAX1(WC1+D+WIND,WC2+D+WIND)+WFEED
C
      IF(YCL.GE.YCLR)YCLR=YCL
C
      RETURN
      END
C
      SUBROUTINE CAP11(WANT,CC,WC,LC,EPSR,THICK)
C THIS SUBROUTINE CALCULATES THE DIMENSIONS OF A METAL FOIL
C CAPACITOR FROM GIVEN MILAR PROPERTIES.  THIS SUBPROGRAM
C DETERMINES THE PARAMETERS OF THE SHUNT-CONNECTED CAPACITORS
C OF TYPE-11 AND RELATED NETWORKS.
C
      REAL LC
C
      C=CC*1E-12
      TH=THICK*2.54E-5
      A=C*TH/(8.854E-12*EPSR)
      A=A/(.0254**2)
C
      WC=WANT
      LC=A/WC
C
      IF(LC.GT.2.*WANT)THEN
         WC=2.*WANT
         LC=A/WC
      ENDIF
C
      RETURN
      END
C
C
      SUBROUTINE CAPAA(FL,WID,CC,EPSR,THICK)
C THIS SUBROUTINE CALCULATES THE DIMENSIONS OF A METAL FOIL
C CAPACITOR FROM GIVEN MILAR PROPERTIES.  THIS SUBPROGRAM
C DETERMINES THE PARAMETERS OF THE SHUNT-CONNECTED CAPACITORS
C OF TYPE-11 AND RELATED NETWORKS.
C
C
      C=CC*1E-12
      TH=THICK*2.54E-5
      A=C*TH/(8.854E-12*EPSR)
      A=A/(.0254**2)
C
      WID=A/FL
C
      RETURN
      END
C
C
      REAL FUNCTION LSTRIP(L,W)
      REAL L
C
      DATA DELTAX/3E-4/,ERROR/1E-3/
C
```

```
C  STATEMENT FUNCTION FOR INDUCTANCE OF A STRIP.
      AINDUCT(X,W)=5.08*X*(LOG(2.+X/W)+.5)
C
C GUESS A VALUE
      X1=L/20.
C
   10 Y1=L-AINDUCT(X1,W)
      Y2=L-AINDUCT(X1+DELTAX,W)
      X2=X1-DELTAX*Y1/(Y2-Y1)
      IF(ABS(X2-X1).LE.ERROR)THEN
         LSTRIP=X2
         RETURN
      ELSE
         X1=X2
         GO TO 10
      ENDIF
C
      END
C
C
C
C ***************************************************************************** C
C
C                  MATCHING NETWORK DESIGN AND LAYOUT
C
C ***************************************************************************** C
C
      SUBROUTINE NET09(XORG,YORG,L1,C2,C3,LANT,WANT,WFEED,THICK,EPSR,
     &   FTHICK,IROT,ANGLE,YCLR,LAMBDA,ILOC,M,NROW)
C
C  THIS SUBROUTINE DRAWS A TYPE-9 NETWORK.
C  THE TYPE-9 NETWORK IS A HIGH-PASS FORM WITH A CAPACITOR IN SERIES WITH
C  EACH ANTENNA ELEMENT, A SHUNT CAPACITOR BETWEEN ANTENNAS AND A SHUNT
C  INDUCTOR ACROSS THE FEEDLINE.  L1 IS THE SHUNT INDUCTOR, C2 IS THE
C  SHUNT INDUCTOR AND C3 IS THE SERIES CAPACITOR.
C
      REAL L1
      REAL LC,LC2,LANT
      REAL LAMBDA
C
      DATA LTOP/1/,LBOT/2/,LBRD/3/
      DATA CLR/.05/
      DATA WIND/.125/,WMIN/.125/
C
C  DETERMINE THE DIMENSIONS OF THE SERIES CAPACITORS.
C
      CALL CAPVAL(WANT,C3,WC,LC,EPSR,THICK)
C
C  EVALUATE SOME CONSTANTS NEEDED LATER
      LC2=LC/2.
      WF2=WFEED/2.
      XANT=LANT-(CLR+WF2+LC2)
C
C   DRAW THE RIGHT-HAND (TOP LAYER) ANTENNA AND SERIES CAPACITOR.
C
      CALL BOX(XORG,YORG,LC+WF2+CLR,WC,3,IROT,XORG,YORG,ANGLE,LTOP)
      CALL BOX(XORG+WF2+CLR,YORG,LC,WC,1,IROT,XORG,YORG,ANGLE,LTOP)
      CALL BOX(XORG+CLR+WF2+LC2,YORG,XANT,WANT,1,IROT,XORG,YORG,
     &         ANGLE,LTOP)
C
C  DRAW THE LEFT-HAND (BOTTOM LAYER) ANTENNA AND SERIES CAPACITOR.
C
      CALL BOX(XORG,YORG,LC+WF2+CLR,WC,1,IROT,XORG,YORG,ANGLE,LBOT)
      CALL BOX(XORG-WF2-CLR,YORG,LC,WC,3,IROT,XORG,YORG,ANGLE,LBOT)
      CALL BOX(XORG-CLR-WF2-LC2,YORG,XANT,WANT,3,IROT,XORG,YORG,
     &         ANGLE,LBOT)
C
C  DRAW THE INDUCTORS
C
```

```
C  CALCULATE AND DRAW THE INDUCTOR ACROSS THE FEEDLINE.
C
      IF(L1.GE.0.)THEN
         CALL SHUNTL(XORG,YORG,WFEED,WIND,FTHICK,WC,LC,
     &               CLR,L1,LAMBDA,1,IROT,ANGLE,YCL)
      ENDIF
C
C  MAKE A VIA
C     CALL VIA(X1+WIND2,Y1,IROT,ANGLE,LVIA)
C
C  CALCULATE AND DRAW THE CAPACITOR ACROSS THE ANTENNA.
C
      IF(C2.GE.0.)THEN
         CALL SHUNTC(XORG,YORG,C2,WC,WFEED,WMIN,
     &               THICK,EPSR,IROT,ANGLE,CLR,2,YCLR)
      ENDIF
C
C  MAKE A VIA
C     CALL VIA(X1-WIND2,Y1,IROT,ANGLE,LVIA)
C
C  KEEP TRACK OF THE SPACE OCCUPIED BY THE SHUNT INDUCTORS SO
C  THAT THE FEEDLINE EXTENSIONS CAN BE MADE LONG ENOUGH TO CLEAR
C  ALL ELEMENTS IN THE MATCHING NETWORKS.
C
      IF(ILOC.EQ.1.AND.M.EQ.NROW.AND.NROW.NE.1)RETURN
      IF(ILOC.EQ.2.AND.M.EQ.1.AND.NROW.NE.1)RETURN
C
      IF(YCL.GE.YCLR)YCLR=YCL
C
      RETURN
      END
C
C
      SUBROUTINE NET10(XORG,YORG,L1,L2,C3,LANT,WANT,WFEED,THICK,EPSR,
     &   FTHICK,IROT,ANGLE,YCLR,LAMBDA,ILOC,M,NROW)
C
C  THIS SUBROUTINE DRAWS A TYPE-10 NETWORK.
C  THE TYPE-10 NETWORK IS A HIGH-PASS FORM WITH A CAPACITOR IN SERIES WITH
C  EACH ANTENNA ELEMENT AND SHUNT INDUCTORS ACROSS THE ANTENNAS AND
C  FEEDLINE, RESPECTIVELY.  L1 AND L2 ARE THE INDUCTORS ACROSS THE FEEDLINE
C  AND ANTENNAS, RESPECTIVELY.  THE TWO CAPACITORS HAVE VALUE C.
C
      REAL L1,L2
      REAL LC,LC2,LANT
      REAL LAMBDA
C
      DATA LTOP/1/,LBOT/2/,LBRD/3/
      DATA CLR/.05/
      DATA WIND/.125/
C
C  DETERMINE THE DIMENSIONS OF THE SERIES CAPACITORS.
C
      CALL CAPVAL(WANT,C3,WC,LC,EPSR,THICK)
C
C  EVALUATE SOME CONSTANTS NEEDED LATER
      LC2=LC/2.
      WF2=WFEED/2.
      XANT=LANT-(CLR+WF2+LC2)
C
C   DRAW THE RIGHT-HAND (TOP LAYER) ANTENNA AND SERIES CAPACITOR.
C
      CALL BOX(XORG,YORG,LC+WF2+CLR,WC,3,IROT,XORG,YORG,ANGLE,LTOP)
      CALL BOX(XORG+WF2+CLR,YORG,LC,WC,1,IROT,XORG,YORG,ANGLE,LTOP)
      CALL BOX(XORG+CLR+WF2+LC2,YORG,XANT,WANT,1,IROT,XORG,YORG,
     &          ANGLE,LTOP)
C
C  DRAW THE LEFT-HAND (BOTTOM LAYER) ANTENNA AND SERIES CAPACITOR.
C
      CALL BOX(XORG,YORG,LC+WF2+CLR,WC,1,IROT,XORG,YORG,ANGLE,LBOT)
```

```
      CALL BOX(XORG-WF2-CLR,YORG,LC,WC,3,IROT,XORG,YORG,ANGLE,LBOT)
      CALL BOX(XORG-CLR-WF2-LC2,YORG,XANT,WANT,3,IROT,XORG,YORG,
     &         ANGLE,LBOT)
C
C DRAW THE INDUCTORS
C
C CALCULATE AND DRAW THE INDUCTOR ACROSS THE FEEDLINE.
C
      IF(L1.GE.0.)THEN
         CALL SHUNTL(XORG,YORG,WFEED,WIND,FTHICK,WC,LC,
     &               CLR,L1,LAMBDA,1,IROT,ANGLE,YCL)
      ENDIF
C
C MAKE A VIA
C     CALL VIA(X1+WIND2,Y1,IROT,ANGLE,LVIA)
C
C CALCULATE AND DRAW THE INDUCTOR ACROSS THE TWO ANTENNAS.
C
      IF(L2.GE.0.)THEN
         CALL SHUNTL(XORG,YORG,WFEED,WIND,FTHICK,WC,LC,
     &               CLR,L2,LAMBDA,2,IROT,ANGLE,YCL)
      ENDIF
C
C MAKE A VIA
C     CALL VIA(X1-WIND2,Y1,IROT,ANGLE,LVIA)
C
C KEEP TRACK OF THE SPACE OCCUPIED BY THE SHUNT INDUCTORS SO
C THAT THE FEEDLINE EXTENSIONS CAN BE MADE LONG ENOUGH TO CLEAR
C ALL ELEMENTS IN THE MATCHING NETWORKS.
C
      IF(ILOC.EQ.1.AND.M.EQ.NROW.AND.NROW.NE.1)RETURN
      IF(ILOC.EQ.2.AND.M.EQ.1.AND.NROW.NE.1)RETURN
C
      IF(YCL.GE.YCLR)YCLR=YCL
C
      RETURN
      END
C
C
      SUBROUTINE NET11(XORG,YORG,C1,C2,L3,LANT,WANT,WFEED,THICK,EPSR,
     &   FTHICK,IROT,ANGLE,YCLR,LAMBDA,ILOC,M,NROW)
C
C THIS SUBROUTINE DRAWS A TYPE-11 NETWORK.
C THE TYPE-11 NETWORK IS A LOW-PASS FORM WITH AN INDUCTOR IN SERIES WITH
C EACH ANTENNA ELEMENT AND SHUNT CAPACITORS ACROSS THE ANTENNAS AND
C FEEDLINE, RESPECTIVELY.  C1 AND C2 ARE THE CAPACITORS ACROSS THE FEEDLINE
C AND ANTENNAS, RESPECTIVELY.  THE TWO INDUCTORS HAVE VALUE L3.
C
      REAL L3,LIND
      REAL LC,LC2,LANT
      REAL LAMBDA,LSTRIP
C
      DATA LTOP/1/,LBOT/2/,LBRD/3/
      DATA CLR/.05/,OL/.05/,WSRIND/.025/,WX/.20/
C
C DETERMINE THE DIMENSIONS OF THE SERIES INDUCTORS.
C
      NOTUSE=LAMBDA
C
      LIND= LSTRIP(L3,WSRIND)
C
C GET THE DIMENSIONS OF THE CAPACITORS ACROSS FEED LINE
C ONE-HALF OF CAPACITANCE ON EACH SIDE
C
      CALL CAP11(WANT,C1/2.,WC,LC,EPSR,THICK)
C
C EVALUATE SOME CONSTANTS NEEDED LATER
      WF2=WFEED/2.
      XANT=LANT-(LIND+WF2+LC)
```

```
C
C  DRAW THE RIGHT-HAND (TOP) ANTENNA AND SERIES INDUCTOR.
C
      CALL BOX(XORG+LC+WF2-OL,YORG,LIND+2.*OL,WSRIND,1,IROT,XORG,
     &         YORG,ANGLE,LTOP)
      CALL BOX(XORG+LIND+LC+WF2,YORG,XANT,WANT,1,IROT,XORG,YORG,
     &         ANGLE,LTOP)
C
C DRAW THE LEFT-HAND (BOTTOM) ANTENNA AND SERIES INDUCTOR.
C
      CALL BOX(XORG-LC-WF2+OL,YORG,LIND+2.*OL,WSRIND,3,IROT,XORG,
     &         YORG,ANGLE,LBOT)
      CALL BOX(XORG-LIND-LC-WF2,YORG,XANT,WANT,3,IROT,XORG,YORG,
     &         ANGLE,LBOT)
C
C DRAW THE CAPACITORS SHUNTING THE FEED LINE.
C
      IF(C1.GT.0.)THEN
C
         CALL BOX(XORG+WF2-OL,YORG,LC+OL,WC,1,IROT,XORG,YORG,
     &            ANGLE,LTOP)
         CALL BOX(XORG+WF2-OL,YORG,LC+OL,WC,1,IROT,XORG,YORG,
     &            ANGLE,LBOT)
         CALL BOX(XORG-WF2+OL,YORG,LC+OL,WC,3,IROT,XORG,YORG,
     &            ANGLE,LTOP)
         CALL BOX(XORG-WF2+OL,YORG,LC+OL,WC,3,IROT,XORG,YORG,
     &            ANGLE,LBOT)
C
      ENDIF
C
C
C CALCULATE AND DRAW THE CAPACITOR BETWEEN THE ANTENNAS.
C
      WC2=WC/2.
      DC=WX+CLR+WC2+WANT/2.-OL
C
      IF(C2.GE.0.)THEN
C
      CALL CAPAA(DC,X,C2,EPSR,THICK)
C
C DRAW THE LEFT-HAND PLATE CONNECTED TO THE ANTENNA (BOTTOM SIDE)
      CALL BOX(XORG-WF2-LC-LIND-X/2.,YORG-WC2-CLR-WX,DC,X,2,IROT,
     &         XORG,YORG,ANGLE,LBOT)
C
C DRAW THE RIGHT-HAND PLATE CONNECTED TO THE ANTENNA (TOP SIDE)
      CALL BOX(XORG+WF2+LC+LIND+X/2.,YORG-WC2-CLR-WX,DC,X,2,IROT,
     &         XORG,YORG,ANGLE,LYOP)
C
C DRAW THE LEFT-HAND PLATE CONNECTED TO THE CROSSOVER (TOP SIDE).
      CALL BOX(XORG-WF2-LC-LIND-X/2.,YORG-WC2-CLR-WX,DC,X,2,IROT,
     &         XORG,YORG,ANGLE,LTOP)
C
C DRAW THE CROSS-OVER CONNECTING THE TOP PLATES ON BOTH SIDES
      CALL BOX(XORG+WF2+LC+LIND+X/2.,YORG-WC2-CLR-WX/2.,
     &         2.*(WF2+LC+LIND+X/2.),WX,3,IROT,XORG,YORG,ANGLE,LBRD)
C
      ENDIF
C
C MAKE A VIA
C     CALL VIA(X1+WIND2,Y1,IROT,ANGLE,LVIA)
C
C KEEP TRACK OF THE SPACE OCCUPIED BY THE SHUNT INDUCTORS SO
C THAT THE FEEDLINE EXTENSIONS CAN BE MADE LONG ENOUGH TO CLEAR
C ALL ELEMENTS IN THE MATCHING NETWORKS.
C
      IF(ILOC.EQ.1.AND.M.EQ.NROW.AND.NROW.NE.1)RETURN
      IF(ILOC.EQ.2.AND.M.EQ.1.AND.NROW.NE.1)RETURN
      YCL=WC2+CLR+WX+WFEED
      IF(YCL.GT.YCLR)YCLR=YCL
```

```fortran
C
      RETURN
      END
C
      SUBROUTINE NET12(XORG,YORG,C1,L2,L3,LANT,WANT,WFEED,THICK,EPSR,
     &   FTHICK,IROT,ANGLE,YCLR,LAMBDA,ILOC,M,NROW)
C
C THIS SUBROUTINE DRAWS A TYPE-12 NETWORK.
C THE TYPE-12 NETWORK IS A LOW-PASS FORM WITH AN INDUCTOR IN SERIES WITH
C EACH ANTENNA ELEMENT, SHUNT CAPACITOR ACROSS THE FEED LINE AND
C A SHUNT INDUCTOR ACROSS THE TWO ANTENNAS.  C1 IS THE CAPACITOR ACROSS
C THE FEEDLINE, L2 IS THE INDUCTOR ACROSS THE ANTENNAS, AND L3 IS THE
C INDUCTOR IN SERIES WITH EACH ANTENNA ELEMENT.
C
      REAL L2,L3,LIND
      REAL LC,LC2,LANT
      REAL LAMBDA,LSTRIP
C
      DATA LTOP/1/,LBOT/2/,LBRD/3/
      DATA CLR/.05/,OL/.05/,WSRIND/.05/,WX/.25/,WIND/.125/
C
C DETERMINE THE DIMENSIONS OF THE SERIES INDUCTORS.
C
      NOTUSE=LAMBDA
C
      LIND= LSTRIP(L3,WSRIND)
C
C GET THE DIMENSIONS OF THE CAPACITORS ACROSS FEED LINE
C ONE-HALF OF CAPACITANCE ON EACH SIDE
C
      CALL CAP11(WANT,C1/2.,WC,LC,EPSR,THICK)
C
C EVALUATE SOME CONSTANTS NEEDED LATER
      WF2=WFEED/2.
      XANT=LANT-(LIND+WF2+LC)
C
C  DRAW THE RIGHT-HAND (TOP) ANTENNA AND SERIES INDUCTOR.
C
      CALL BOX(XORG+LC+WF2-OL,YORG,LIND+2.*OL,WSRIND,1,IROT,XORG,
     &          YORG,ANGLE,LTOP)
      CALL BOX(XORG+LIND+LC+WF2,YORG,XANT,WANT,1,IROT,XORG,YORG,
     &          ANGLE,LTOP)
C
C DRAW THE LEFT-HAND (BOTTOM LAYER) ANTENNA AND SERIES INDUCTOR.
C
      CALL BOX(XORG-LC-WF2+OL,YORG,LIND+2.*OL,WSRIND,3,IROT,XORG,
     &          YORG,ANGLE,LBOT)
      CALL BOX(XORG-LIND-LC-WF2,YORG,XANT,WANT,3,IROT,XORG,YORG,
     &          ANGLE,LBOT)
C
C DRAW THE CAPACITORS SHUNTING THE FEED LINE.
C
      CALL BOX(XORG+WF2-OL,YORG,LC+OL,WC,1,IROT,XORG,YORG,
     &          ANGLE,LTOP)
      CALL BOX(XORG+WF2-OL,YORG,LC+OL,WC,1,IROT,XORG,YORG,
     &          ANGLE,LBOT)
      CALL BOX(XORG-WF2+OL,YORG,LC+OL,WC,3,IROT,XORG,YORG,
     &          ANGLE,LTOP)
      CALL BOX(XORG-WF2+OL,YORG,LC+OL,WC,3,IROT,XORG,YORG,
     &          ANGLE,LBOT)
C
C CALCULATE AND DRAW THE INDUCTOR BETWEEN THE ANTENNAS.
C
      OW2=WF2+LC+LIND+WIND
      OW=2.*OW2
      WC2=WC/2.
      WANT2=WANT/2.
      WIND2=WIND/2.
C
```

```
      IF(L2.GT.0.)THEN
      CALL INDVAL(L2,OW,D,LAMBDA,WIND,FTHICK)
C
      IF(WC2-WANT2.GT.D)THEN
         D=WC2-WANT2+CLR
      ENDIF
C
C
      CALL BOX(XORG-OW2+WIND2,YORG-WANT2+OL,D+WIND+OL,WIND,4,IROT,
     &         XORG,YORG,ANGLE,LBOT)
      CALL BOX(XORG+OW2-WIND2,YORG-WANT2+OL,D+WIND+OL,WIND,4,IROT,
     &         XORG,YORG,ANGLE,LTOP)
      CALL BOX(XORG-OW2,YORG-WANT2-D-WIND2,OW,WIND,1,IROT,
     &         XORG,YORG,ANGLE,LBRD)
C
      ENDIF
C
C  MAKE A VIA
C     CALL VIA(X1+WIND2,Y1,IROT,ANGLE,LVIA)
C
C  KEEP TRACK OF THE SPACE OCCUPIED BY THE SHUNT INDUCTORS SO
C  THAT THE FEEDLINE EXTENSIONS CAN BE MADE LONG ENOUGH TO CLEAR
C  ALL ELEMENTS IN THE MATCHING NETWORKS.
C
      IF(ILOC.EQ.1.AND.M.EQ.NROW.AND.NROW.NE.1)RETURN
      IF(ILOC.EQ.2.AND.M.EQ.1.AND.NROW.NE.1)RETURN
      YCL=WC2+CLR+WX+WFEED
      IF(YCL.GT.YCLR)YCLR=YCL
C
      RETURN
      END
C
C
      SUBROUTINE NET13(XORG,YORG,C1,L2,C3,LANT,WANT,WFEED,THICK,EPSR,
     &   FTHICK,IROT,ANGLE,YCLR,LAMBDA,ILOC,M,NROW)
C
C  THIS SUBROUTINE DRAWS A TYPE-13 NETWORK.
C  THE TYPE-13 NETWORK IS A HIGH-PASS FORM WITH A CAPACITOR IN SERIES WITH
C  EACH ANTENNA ELEMENT, A SHUNT CAPACITOR ACROSS THE FEEDLINE, AND
C  A SHUNT INDUCTOR BETWEEN ANTENNAS.  C1 IS THE CAPACITOR ACROSS THE
C  FEEDLINE, L2 IS THE INDUCTOR BETWEEN THE ANTENNAS, AND C3 IS
C  THE CAPACITOR IN SERIES WITH EACH ANTENNA.
C
      REAL L2
      REAL LC,LC2,LANT
      REAL LAMBDA
C
      DATA LTOP/1/,LBOT/2/,LBRD/3/
      DATA CLR/.05/
      DATA WIND/.125/,WMIN/.125/
C
C  DETERMINE THE DIMENSIONS OF THE SERIES CAPACITORS.
C
      CALL CAPVAL(WANT,C3,WC,LC,EPSR,THICK)
      YCL=WC/2.+WIND
C
C  EVALUATE SOME CONSTANTS NEEDED LATER
      LC2=LC/2.
      WF2=WFEED/2.
      XANT=LANT-(CLR+WF2+LC2)
C
C   DRAW THE RIGHT-HAND (TOP LAYER) ANTENNA AND SERIES CAPACITOR.
C
      CALL BOX(XORG,YORG,LC+WF2+CLR,WC,3,IROT,XORG,YORG,ANGLE,LTOP)
      CALL BOX(XORG+WF2+CLR,YORG,LC,WC,1,IROT,XORG,YORG,ANGLE,LTOP)
      CALL BOX(XORG+CLR+WF2+LC2,YORG,XANT,WANT,1,IROT,XORG,YORG,
     &         ANGLE,LTOP)
C
C  DRAW THE LEFT-HAND (BOTTOM LAYER) ANTENNA AND SERIES CAPACITOR.
```

```
C
      CALL BOX(XORG,YORG,LC+WF2+CLR,WC,1,IROT,XORG,YORG,ANGLE,LBOT)
      CALL BOX(XORG-WF2-CLR,YORG,LC,WC,3,IROT,XORG,YORG,ANGLE,LBOT)
      CALL BOX(XORG-CLR-WF2-LC2,YORG,XANT,WANT,3,IROT,XORG,YORG,
     &          ANGLE,LBOT)
C
C DRAW THE INDUCTORS
C
C CALCULATE AND DRAW THE INDUCTOR ACROSS THE ANTENNA.
C
      IF(L2.GE.0.)THEN
         CALL SHUNTL(XORG,YORG,WFEED,WIND,FTHICK,WC,LC,
     &               CLR,L2,LAMBDA,2,IROT,ANGLE,YCL)
      ENDIF
C
C MAKE A VIA
C     CALL VIA(X1+WIND2,Y1,IROT,ANGLE,LVIA)
C
C CALCULATE AND DRAW THE CAPACITOR ACROSS THE FEED LINE.
C
      IF(C1.GE.0.)THEN
         CALL SHUNTC(XORG,YORG,C1,WC,WFEED,WMIN,
     &               THICK,EPSR,IROT,ANGLE,CLR,1,YCL)
      ENDIF
C
C MAKE A VIA
C     CALL VIA(X1-WIND2,Y1,IROT,ANGLE,LVIA)
C
C KEEP TRACK OF THE SPACE OCCUPIED BY THE SHUNT INDUCTORS SO
C THAT THE FEEDLINE EXTENSIONS CAN BE MADE LONG ENOUGH TO CLEAR
C ALL ELEMENTS IN THE MATCHING NETWORKS.
C
C     WRITE(0,*)'IN NET13; YCL,YCLR: ',YCL,YCLR
C
      IF(ILOC.EQ.1.AND.M.EQ.NROW.AND.NROW.NE.1)RETURN
      IF(ILOC.EQ.2.AND.M.EQ.1.AND.NROW.NE.1)RETURN
      IF(YCL.GE.YCLR)YCLR=YCL
C
      RETURN
      END
C
C
      SUBROUTINE NET14(XORG,YORG,L1,C2,L3,LANT,WANT,WFEED,THICK,EPSR,
     &   FTHICK,IROT,ANGLE,YCLR,LAMBDA,ILOC,M,NROW)
C
C THIS SUBROUTINE DRAWS A TYPE-14 NETWORK.
C THE TYPE-14 NETWORK IS A LOW-PASS FORM WITH AN INDUCTOR IN SERIES WITH
C EACH ANTENNA ELEMENT, A SHUNT INDUCTOR ACROSS THE FEEDLINE, AND A
C SHUNT CAPACITOR ACROSS THE ANTENNAS.  THE INDUCTOR ACROSS THE FEEDLINE
C HAS VALUE L1, THE CAPACITOR C2 AND THE INDUCTOR IN SERIES WITH EACH
C ELEMENT IS L3.  IF THE ELEMENT VALUE IS ZERO OR NEGATIVE, SHUNT ELEMENTS
C ARE NOT DRAWN.
C
      REAL L1,L3,LIND
      REAL LC,LC2,LANT
      REAL LAMBDA,LSTRIP
C
      DATA LTOP/1/,LBOT/2/,LBRD/3/
      DATA CLR/.05/,OL/.05/,WSRIND/.025/,WX/.20/,WIND/.125/
C
C IF L1 IS POSITIVE, DRAW THE INDUCTTOR SHUNTING THE FEED LINE.
      IF(L1.GE.0.)THEN
C
C LOGIC TO DECIDE WIDTH OF FEEDLINE CONNECTION
C
         IF(L1.GT.0..AND.L1.LT.10.)THEN
            OW2=0.25
             H=0.25
         ELSEIF(L1.GE.10..AND.L1.LT.50.)THEN
```

```
              OW2=0.375
               H=0.375
           ELSE
              OW2=0.75
               H=0.75
           ENDIF
C
C  DRAW FEED-LINE CONNECTION
C
        CALL BOX(XORG,YORG,OW2,H,3,IROT,XORG,YORG,
     &          ANGLE,LBOT)
        CALL BOX(XORG,YORG,OW2,H,1,IROT,XORG,YORG,
     &          ANGLE,LTOP)
C
        WIND2=WIND/2.
        H2=H/2.
C
        CALL INDVAL(L1,2.*OW2,D,LAMBDA,WIND,FTHICK)
C
        CALL BOX(XORG-OW2+WIND2,YORG+H2-OL,D+WIND+OL,WIND,2,IROT,
     &          XORG,YORG,ANGLE,LBOT)
        CALL BOX(XORG+OW2-WIND2,YORG+H2-OL,D+WIND+OL,WIND,2,IROT,
     &          XORG,YORG,ANGLE,LTOP)
        CALL BOX(XORG-OW2,YORG+H2+D+WIND2,2.*OW2,WIND,1,IROT,
     &          XORG,YORG,ANGLE,LBRD)
C
C
        ENDIF
C
C  DETERMINE THE DIMENSIONS OF THE SERIES INDUCTORS.
C
        LIND= LSTRIP(L3,WSRIND)
C
C  EVALUATE SOME CONSTANTS NEEDED LATER
        WF2=WFEED/2.
        XANT=LANT-(LIND+WF2+OW2)
C
C   DRAW THE RIGHT-HAND (TOP) ANTENNA AND SERIES INDUCTOR.
C
        CALL BOX(XORG+OW2+WF2-OL,YORG,LIND+2.*OL,WSRIND,1,IROT,XORG,
     &          YORG,ANGLE,LTOP)
        CALL BOX(XORG+LIND+OW2+WF2,YORG,XANT,WANT,1,IROT,XORG,YORG,
     &          ANGLE,LTOP)
C
C  DRAW THE LEFT-HAND (BOTTOM) ANTENNA AND SERIES INDUCTOR.
C
        CALL BOX(XORG-OW2-WF2+OL,YORG,LIND+2.*OL,WSRIND,3,IROT,XORG,
     &          YORG,ANGLE,LBOT)
        CALL BOX(XORG-LIND-OW2-WF2,YORG,XANT,WANT,3,IROT,XORG,YORG,
     &          ANGLE,LBOT)
C
C
C  IF C2 IS POSITIVE, CALCULATE AND DRAW THE CAPACITOR BETWEEN THE ANTENNAS.
C
        IF(C2.GT.0.)THEN
C
        DC=WX+CLR+H2+WANT/2.-OL
C
        CALL CAPAA(DC,X,C2,EPSR,THICK)
C
C  DRAW THE LEFT-HAND PLATE CONNECTED TO THE ANTENNA (BOTTOM SIDE)
        CALL BOX(XORG-WF2-OW2-LIND-X/2.,YORG-H2-CLR-WX,DC,X,2,IROT,
     &          XORG,YORG,ANGLE,LBOT)
C
C  DRAW THE RIGHT-HAND PLATE CONNECTED TO THE ANTENNA (TOP SIDE)
        CALL BOX(XORG+WF2+OW2+LIND+X/2.,YORG-H2-CLR-WX,DC,X,2,IROT,
     &          XORG,YORG,ANGLE,LYOP)
C
C  DRAW THE LEFT-HAND PLATE CONNECTED TO THE CROSSOVER (TOP SIDE).
```

```
      CALL BOX(XORG-WF2-OW2-LIND-X/2.,YORG-H2-CLR-WX,DC,X,2,IROT,
     &         XORG,YORG,ANGLE,LTOP)
C
C DRAW THE CROSS-OVER CONNECTING THE TOP PLATES ON BOTH SIDES
      CALL BOX(XORG+WF2+OW2+LIND+X/2.,YORG-H2-CLR-WX/2.,
     &         2.*(WF2+OW2+LIND+X/2.),WX,3,IROT,XORG,YORG,ANGLE,LBRD)
C
      ENDIF
C
C
C MAKE A VIA
C     CALL VIA(X1+WIND2,Y1,IROT,ANGLE,LVIA)
C
C KEEP TRACK OF THE SPACE OCCUPIED BY THE SHUNT INDUCTORS SO
C THAT THE FEEDLINE EXTENSIONS CAN BE MADE LONG ENOUGH TO CLEAR
C ALL ELEMENTS IN THE MATCHING NETWORKS.
C
      IF(ILOC.EQ.1.AND.M.EQ.NROW.AND.NROW.NE.1)RETURN
      IF(ILOC.EQ.2.AND.M.EQ.1.AND.NROW.NE.1)RETURN
      YCL=WC2+CLR+WX+WFEED
      IF(YCL.GT.YCLR)YCLR=YCL
C
      RETURN
      END
C
C
      SUBROUTINE NET15(XORG,YORG,C1,C2,C3,LANT,WANT,WFEED,THICK,EPSR,
     &   FTHICK,IROT,ANGLE,YCLR,LAMBDA,ILOC,M,NROW)
C
C Written 10/04/89
C THIS SUBROUTINE DRAWS A TYPE-15 NETWORK.
C THE TYPE-15 NETWORK IS A PI-TYPE NETWORK WITH ONLY CAPACITORS.
C A CAPACITOR IS IN SERIES WITH EACH ANTENNA AND CAPACITORS ARE
C CONNECTED ACROSS THE FEEDLINE AND ANTENNAS, RESPECTIVELY.
C C1 AND C2 ARE CONNECTED ACROSS THE FEEDLINE AND CAPACITORS, RESPECTIVELY,
C WHILE C3 IS THE SERIES CAPACITOR.
C
      REAL LC,LC2,LANT
      REAL LAMBDA
C
      DATA LTOP/1/,LBOT/2/,LBRD/3/
      DATA CLR/.05/
      DATA WIND/.125/,WMIN/.125/
C
C DETERMINE THE DIMENSIONS OF THE SERIES CAPACITORS.
C
      CALL CAPVAL(WANT,C3,WC,LC,EPSR,THICK)
C
C EVALUATE SOME CONSTANTS NEEDED LATER
      LC2=LC/2.
      WF2=WFEED/2.
      XANT=LANT-(CLR+WF2+LC2)
C
C  DRAW THE RIGHT-HAND (TOP LAYER) ANTENNA AND SERIES CAPACITOR.
C
      CALL BOX(XORG,YORG,LC+WF2+CLR,WC,3,IROT,XORG,YORG,ANGLE,LTOP)
      CALL BOX(XORG+WF2+CLR,YORG,LC,WC,1,IROT,XORG,YORG,ANGLE,LTOP)
      CALL BOX(XORG+CLR+WF2+LC2,YORG,XANT,WANT,1,IROT,XORG,YORG,
     &         ANGLE,LTOP)
C
C DRAW THE LEFT-HAND (BOTTOM LAYER) ANTENNA AND SERIES CAPACITOR.
C
      CALL BOX(XORG,YORG,LC+WF2+CLR,WC,1,IROT,XORG,YORG,ANGLE,LBOT)
      CALL BOX(XORG-WF2-CLR,YORG,LC,WC,3,IROT,XORG,YORG,ANGLE,LBOT)
      CALL BOX(XORG-CLR-WF2-LC2,YORG,XANT,WANT,3,IROT,XORG,YORG,
     &         ANGLE,LBOT)
C
C
C CALCULATE AND DRAW THE CAPACITOR ACROSS THE FEED LINE.
```

```
C
      IF(C1.GE.0.)THEN
         CALL SHUNTC(XORG,YORG,C1,WC,WFEED,WMIN,
     &               THICK,EPSR,IROT,ANGLE,CLR,1,YCLR)
      ENDIF
C
C MAKE A VIA
C     CALL VIA(X1-WIND2,Y1,IROT,ANGLE,LVIA)
C
C CALCULATE AND DRAW THE CAPACITOR ACROSS THE ANTENNA.
C
      IF(C2.GE.0.)THEN
         CALL SHUNTC(XORG,YORG,C2,WC,WFEED,WMIN,
     &               THICK,EPSR,IROT,ANGLE,CLR,2,YCLR)
      ENDIF
C
C KEEP TRACK OF THE SPACE OCCUPIED BY THE SHUNT INDUCTORS SO
C THAT THE FEEDLINE EXTENSIONS CAN BE MADE LONG ENOUGH TO CLEAR
C ALL ELEMENTS IN THE MATCHING NETWORKS.
C
      IF(ILOC.EQ.1.AND.M.EQ.NROW.AND.NROW.NE.1)RETURN
      IF(ILOC.EQ.2.AND.M.EQ.1.AND.NROW.NE.1)RETURN
      IF(YCL.GE.YCLR)YCLR=YCL
C
      RETURN
      END
C
C
      SUBROUTINE NET16(XORG,YORG,L1,L2,L3,LANT,WANT,WFEED,THICK,EPSR,
     &   FTHICK,IROT,ANGLE,YCLR,LAMBDA,ILOC,M,NROW)
C
C THIS SUBROUTINE DRAWS A TYPE-16 NETWORK.
C THE TYPE-16 NETWORK IS A PI-NEWTORK WITH ONLY INDUCTORS.
C L1 IS THE INDUCTOR ACROSS THE FEEDLINE, L2 IS THE INDUCTOR BETWEEN
C ANTENNAS, AND L3 IS THE INDUCTOR IN SERIES WITH EACH ANTENNA.
C IF L1 OR L2 ARE NEGATIVE, THE CORRESPONDING ELEMENT IS NOT DRAWN.
C
      REAL L1,L2,L3,LIND
      REAL LC,LC2,LANT
      REAL LAMBDA,LSTRIP
C
      DATA LTOP/1/,LBOT/2/,LBRD/3/
      DATA CLR/.05/,OL/.05/,WSRIND/.025/,WX/.20/,WIND/.125/
C
C IF L1 IS POSITIVE, DRAW THE INDUCTTOR SHUNTING THE FEED LINE.
      IF(L1.GE.0.)THEN
C
C LOGIC TO DECIDE WIDTH OF FEEDLINE CONNECTION
C
         IF(L1.GT.0..AND.L1.LT.10.)THEN
            OW2=0.25
             H=0.25
         ELSEIF(L1.GE.10..AND.L1.LT.50.)THEN
            OW2=0.375
             H=0.375
         ELSE
            OW2=0.75
             H=0.75
         ENDIF
C
C DRAW FEED-LINE CONNECTION
C
      CALL BOX(XORG,YORG,OW2,H,3,IROT,XORG,YORG,
     &         ANGLE,LBOT)
      CALL BOX(XORG,YORG,OW2,H,1,IROT,XORG,YORG,
     &         ANGLE,LTOP)
C
      WIND2=WIND/2.
      H2=H/2.
```

```
      C
              CALL INDVAL(L1,2.*OW2,D,LAMBDA,WIND,FTHICK)
      C
              CALL BOX(XORG-OW2+WIND2,YORG+H2-OL,D+WIND+OL,WIND,2,IROT,
             &         XORG,YORG,ANGLE,LBOT)
              CALL BOX(XORG+OW2-WIND2,YORG+H2-OL,D+WIND+OL,WIND,2,IROT,
             &         XORG,YORG,ANGLE,LTOP)
              CALL BOX(XORG-OW2,YORG+H2+D+WIND2,2.*OW2,WIND,1,IROT,
             &         XORG,YORG,ANGLE,LBRD)
      C
      C
              ENDIF
      C
      C  DETERMINE THE DIMENSIONS OF THE SERIES INDUCTORS.
      C
              LIND= LSTRIP(L3,WSRIND)
      C
      C  EVALUATE SOME CONSTANTS NEEDED LATER
              WF2=WFEED/2.
              XANT=LANT-(LIND+WF2+OW2)
      C
      C   DRAW THE RIGHT-HAND (TOP) ANTENNA AND SERIES INDUCTOR.
      C
              CALL BOX(XORG+OW2+WF2-OL,YORG,LIND+2.*OL,WSRIND,1,IROT,XORG,
             &         YORG,ANGLE,LTOP)
              CALL BOX(XORG+LIND+OW2+WF2,YORG,XANT,WANT,1,IROT,XORG,YORG,
             &         ANGLE,LTOP)
      C
      C  DRAW THE LEFT-HAND (BOTTOM) ANTENNA AND SERIES INDUCTOR.
      C
              CALL BOX(XORG-OW2-WF2+OL,YORG,LIND+2.*OL,WSRIND,3,IROT,XORG,
             &         YORG,ANGLE,LBOT)
              CALL BOX(XORG-LIND-OW2-WF2,YORG,XANT,WANT,3,IROT,XORG,YORG,
             &         ANGLE,LBOT)
      C
      C
      C
      C  CALCULATE AND DRAW THE INDUCTOR BETWEEN THE ANTENNAS.
      C
              OW2=WF2+LC+LIND+WIND
              OW=2.*OW2
              WC2=WC/2.
              WANT2=WANT/2.
              WIND2=WIND/2.
      C
              IF(L2.GT.0.)THEN
              CALL INDVAL(L2,OW,D,LAMBDA,WIND,FTHICK)
      C
              IF(WC2-WANT2.GT.D)THEN
                 D=WC2-WANT2+CLR
              ENDIF
      C
      C
              CALL BOX(XORG-OW2+WIND2,YORG-WANT2+OL,D+WIND+OL,WIND,4,IROT,
             &         XORG,YORG,ANGLE,LBOT)
              CALL BOX(XORG+OW2-WIND2,YORG-WANT2+OL,D+WIND+OL,WIND,4,IROT,
             &         XORG,YORG,ANGLE,LTOP)
              CALL BOX(XORG-OW2,YORG-WANT2-D-WIND2,OW,WIND,1,IROT,
             &         XORG,YORG,ANGLE,LBRD)
      C
              ENDIF
      C
      C
      C  MAKE A VIA
      C       CALL VIA(X1ZWIND2,Y1,IROT,ANGLE,LVIA)
      C
      C  KEEP TRACK OF THE SPACE OCCUPIED BY THE SHUNT INDUCTORS SO
      C  THAT THE FEEDLINE EXTENSIONS CAN BE MADE LONG ENOUGH TO CLEAR
      C  ALL ELEMENTS IN THE MATCHING NETWORKS.
```

```
C
      IF(ILOC.EQ.1.AND.M.EQ.NROW.AND.NROW.NE.1)RETURN
      IF(ILOC.EQ.2.AND.M.EQ.1.AND.NROW.NE.1)RETURN
      YCL=WC2+CLR+WX+WFEED
      IF(YCL.GT.YCLR)YCLR=YCL
C
      RETURN
      END
```